DISC OBJECT MODULE

LIBRARY MAINTENANCE

ROUTINE FOR TDOS

DOMLMR

TABLE OF CONTENTS

TABLE OF CONTENTS

MAINTENANCE SECTION

APPENDIX	CONTENTS	PAGE
A	Input Record to Output Record Conversion	
	General	A-1
	Object Module Directory Block $(00)_{16}$	A-2
	Index Block $(01)_{16}$	A-4
	Object Module Descriptor Block $(02)_{16}$	A-7
	EXTRN Block $(03)_{16}$	A-8
	Text Block $(04)_{16}$	A-9
	Modifier Block $(05)_{16}$	A-10
B	DOMLMR Control Record	
	Format	B-1
	Format Example	B-2
	Disc address Translation Example	B-3

Title: Disc Object Module Library Maintenance Routine (DOMLMR)

Author: William R. Jones - RCA
Cleveland District
Suite 1605
250 East Broad Street
Columbus, Ohio 43215
(614) 464-0400

Date: September 21, 1970

Description: DOMLMR maintains an Object Module Library (OML), on a 70/564 or 70/590 disc unit, in a format acceptable to the RCA Linkage Editor (LNKEDT) utility. DOMLMR will add, delete, and replace individual or groups of modules without the need to replace the entire disc resident library. Because of a difference in format between a disc OML created by the Call Library Transcriber Routine (CLTR) and DOMLMR, this utility will not maintain the master OML. Any attempt to update the master OML with DOMLMR will result in an error message and program termination.

The disc area used by DOMLMR must begin at cylinder 1, track zero, and may extend thru cylinder 200 on a 70/564 or 70/590. The minimum allocation is two cylinders. After allocation of the disc area with the RCA utility Random Access Storage Allocator (RAALLR), the area is preformatted by the PREOML program provided with this package. The allocated disc area may be increased at any time by running RAALLR, followed by PREOML.

The input to DOMLMR is an OML - the output from the Object Module Library Update routine (OMLU). Three standard OMLU control cards may be used to convert an Object Module File (OMF) from a language translator (SYSUT1) to an OML. These three control cards will convert a single or multiple module SYSUT1.

Deletion of modules from the disc OML is accomplished via a console message. The program will request the names of modules to delete when it is run outside of monitor. Up to an 80 character message is accepted from the console which consists of variable length (1 to 8 characters) module names separated by commas. In addition to module names, two special meaning operands may be entered:

$STOP - If this operand is used, it must be the last entry. This operand specifies no input tape. When it is recognized in the parameter string, the program goes to normal termination.

$PRINT- This operand causes a listing of the names of all the modules contained in the OML.

When existing modules are deleted from the OML, all the records used by that module are released for subsequent use. For this reason, no file re-organization is required. When a later version of a module is transcribed to the OML, the module is technically deleted from the file and then re-entered as an addition. This allows the size of the module to change without regard for the original area occupied.

DOMLMR will not properly catalog a module containing a sort due to the fact that OMLU does not convert all of the required information contained on SYSUT1. If an attempt is made to convert and catalog a module containing a sort, OMLU will reflect the omitted information in on error listing, produce an OMF and DOMLMR will catalog OMLU's output without warning. Such a cataloged module will not link properly and should be deleted from the disc OML.

Backup for an OML maintained by DOMLMR is provided by the RCA utility Disc Dump and Reload (DDRL). Once a module is cataloged into the OML, there is no way to retrieve it

except via LNKEDT. There is no facility to extract any particular module from the file for subsequent storage on tape or punched cards.

DOMLMR and PREOML contain a constant which reflects the volume serial number of the disc containing the OML. For this reason, no run time parameters are required for proper device assignment. The only program change required is setting these values which are at a tag 'NAME3.'

The user must bear in mind that subsequent software releases may render certain cataloged modules unusable due to changes to the language translators that modify generated coding. The main area of concern is code generated for FCP and by COBOL for the 'CALL' and 'RETURN' verbs. As changes of this type are seldom, they do not pose a serious problem for DOMLMR users.

The storage capacity is:

70/590 - 4 records per track -
1688 bytes per records
70/564 - 3 records per track -
1129 bytes per record

Cylinder one tracks zero thru eight contain the Object Module Directory - one 16 byte entry for each module contained in the library:

70/590 - 105 entries per record
3778 entries maximum (note 1)
70/564 - 70 entries per record
1888 entries maximum (notes 1 and 2)

Cylinder one, track nine, contains the DOMLMR control record.

Cylinder one, tracks eleven thru nineteen, for 70/590 are not used.

Cylinder two thru the last cylinder in the extent contain the data records or actual modules. Each module will require at least two

of these records for storage. The data records are fixed length of 1688 or 1129 bytes to facilitate update writes to the disc; however, the data contained in them is variable length.

Note 1 - Two of the entries in this area are control entries, the first one and the last one.

Note 2 - The 70/564 does not use the last byte of the 1129 byte directory record.

The PREOML and DOMLMR routines are programmed for both 70/590 and 70/564 support. The device used is determined by the assignment made. No program changes are required.

If DOMLMR terminates abnormally, it cannot be run again until the disc extent has been re-established by DDRL. Access is still possible by LNKEDT.

Equipment:

One tape station for input (SYSUT2).
One disc drive with the 'Object Module Lib' extent.
One printer if the $PRINT option is used in reply to the 'MODULES TO DELETE ?' message.

Memory Required:

Approximately 17,000 bytes.

Source Language:

Assembly

Input Data Format:

The input to DOMLMR is an OML formatted as described in the TOS Utilities Manual 70-35-302. Parameter cards are not required for this program. The names of modules to delete from the disc OML are accepted from the console when the utility is run outside of monitor. The 'Operating Procedures' section of this narrative describes the format and options for the console request.

Output Data Format: The output of DOMLMR is described in appendix A of this narrative. Appendix A also corrolates the input fields that are used with the output format on disc.

Timing: The amount of time required to read the tape and transcribe it to disc.

Remarks:

1. PREOML and DOMLMR are available as source card decks.

2. I would be glad to answer any questions regarding DOMLMR for any interested party.

Operating Procedures (Initialization):

1. Allocate disc area for a file named 'OBJECT MODULE LIB'. This area must begin on cylinder one, track zero, and consist of at least two full cylinders. Only full cylinder allocation is acceptable to the DOMLMR system. The extent must consist of contiguous cylinders.

2. Change the volume serial number in the DC entry called 'NAME3' of both the DOMLMR and PREOML programs to reflect the volume serial numbers allocated in Step 1.

3. Assemble and run the PREOML program. The 'STOP CONDITIONS' portion of this narrative explains the typeouts produced. No parameters are required.

4. Assemble and transcribe the DOMLMR program to the master 'PGMLIB' on the executive disc. The system is now ready for use.

(Extent Change): From time to time it may be necessary to increase the size of the extent allocated as the OML. This can be accomplished by running RAALLR to de-allocate (not purge) the extent and then to re-establish it. Following the RAALLR run, PREOML must be run to update the control record to reflect the change prior to running DOMLMR. The 'STOP CONDITIONS' portion of this narrative explains the typeouts produced by PREOML. It is recommended that DDRL be run to provide a tape backup for the extent immediately after PREOML has been run.

(General): DOMLMR may be run as part of a monitor job stream or independently under the TDOS executive. Device assignment is automatic for the required disc device.

INPUT: An Object Module File (OMF) which is SYSUT2 out of the OMLU utility. This assignment is optional - see 'MODULES TO DELETE ?' message.

OUTPUT: Disc resident OMF.
Printer (SYSLST) - optional - See 'MODULES TO DELETE ?' message.

PARAMETERS: No punched card parameters - see 'MODULES TO DELETE ?' message for console parameter format and options.

MESSAGES FROM PREOML

MESSAGE	MEANING	ACTION
EXTENT NOT ALLOCATED IN FULL CYLINDERS	The disc extent established by RAALLR for use with DOMLMR must be allocated in increments of full cylinders. Run RAALLR again to correct the problem and retry PREOML.	None - program terminates.
EXTENT DOES NOT BEGIN AT CYLINDER 1 HEAD 0	The disc extent established by RAALLR for use with DOMLMR must begin at that location. Run RAALLR again to correct the problem and retry PREOML.	None - program terminates.
NEW ENDING CYL NO IS = OR < OLD ENDING CYL NO	The new allocation decreased the size of the extent rather than increasing it, or, the new area was not allocated contiguous to the old area.	None - program terminates. Run RAALLR again to correct the problem and re-run PREOML.
RECORD READ WAS NOT CONTROL RECORD	See 'MESSAGES FROM DOMLMR ROUTINE'	
IS THIS RUN TO CHANGE EXTENTS? (Y OR N)	Self-explanatory.	Response of 'N' causes the entire object module library area to be pre-formatted with dummy records. Response of 'Y' causes pre-formatting of the new area only and updating of the control record.

MESSAGES FROM DOMLMR ROUTINE

MESSAGE	MEANING	ACTION
TAPE MOUNTED IS NOT AN OML	Input to DOMLMR must be an OML (output of OMLU).	None - program terminates. Correct problem and restart the program.
RECORD READ WAS NOT CONTROL RECORD	A control record is maintained in the first cylinder of the 'OBJECT MODULE LIB' extent. A record was read from that area but it is not the control record. Area may not have been pre-formatted by the PREOML routine or an attempt was made to update the master OML with DOMLMR.	None - TERMD
MODULES TO DELETE ?	Program is requesting the name or names of modules to be removed from the OML. This message is typed only if the program is run outside of monitor.	If you do not wish to delete any modules reply 'EOT' else, enter 1 to 8 character module names in any sequence separated by a comma. If you wish to delete modules only and have no input OML, enter $STOP as the last module name. If you wish to print a listing of all module names contained in the library enter $PRINT as a module name. (Continued)

MESSAGE	MEANING	ACTION
MODULES TO DELETE ? ...CONT..		The $PRINT operand may be used along with module names for deletion and/or with the $STOP operand. It is recommended that the $PRINT operand be used as the last operand entered or immediately preceeding the $STOP operand. Up to 80 characters, including commas, may be entered in reply to this message.
MODULE XXXXXXXX NOT IN LIBRARY	XXXXXXXX represents a module name designated to be deleted. The module name was not found in the directory in cylinder 1.	none - program continues
CAUTION ... OVER 95% OF DATA AREA IS USED UP >>>>>>>>	The disc extent must be extended or unused modules must be removed to provide more room.	None - program continues
OUT OF ROOM MODULE XXXXXXXX NOT TRANSCRIBED	While searching the record table, the end of the table was reached without locating an open record.	None - program terminates. Reload program and delete module indicated. Delete additional modules if possible or else extend disc file area.
NUMBER OF UNUSED DATA RECORDS AT START OF THIS RUN IS XXXXX	This message will be typed the first time the program is run in a day. If this figure is low, delete unused modules or	None - program continues.

MESSAGE	MEANING	ACTION
NUMBER OF UNUSED DATA RECORDS AT START OF THIS RUN IS XXXXX (Continued)	extend the file size upon completion of the run. This message is also typed if programs are to be deleted, in which case the message is for information only.	
NUMBER OF UNUSED DATA RECORDS AT END OF DELETION IS XXXXX	Information only. Message is typed after completion of the delete logic. The difference between this message and the previous message will show the number of records made available as a result of the deletion.	None - program continues.
EXTENT HAS CHANGED - RUN PREOML AND RESTART	RAALLR utility was run to change the extent size on disc. PREOML must now be run to update the disc resident control record prior to accessing the file with DOMLMR.	Program terminates - run PREOML and restart.
FILE NOT CLOSED IN PREVIOUS RUN - THIS RUN TERMINATED.	Reconstruct disc extent from DDRL backup and restart program.	Program terminates - No action taken.

MESSAGES FROM DOMLMR AND PREOML	MEANING	ACTION
DISC I/O ERROR - TYPE R FOR RETRY (Note 1)	I/O to disc has been issued twice without success.	Try at least one more time before giving up.
MOUNT VSN XXXXXX - RUN OLC - TYPE C TO CONTINUE	Required volume, as indicated by XXXXXX is not on line.	Self explanatory.
FOLLOWING FILE NOT FOUND (The line following the message will be the 6 byte volume serial number followed by a 44 byte file name.)	The VTOC of the specified disc was searched and the file was not found. PROGRAMMER: Check the length and contents of the &EMTX operand of the GETEM macro. OPERATOR: May have 2 disc packs on line with same volume serial number.	None - TERMD

Note 1 - This message is also used by the program outside of the macro.

Programmer Considerations for Use of the Disc Object Module Library Maintenance Routine (DOMLMR)

The DOMLMR program accepts an Object Module Library (OML) tape as input and transcribes every module on the tape to disc. The output of a program translator (SYSUT1) is an Object Module File (OMF). To convert an OMF to an OML for use with DOMLMR, it is necessary to run the Object Module Library Update (OMLU) routine. The input to OMLU is a SYSUT1 tape, the output is SYSUT2. The following example shows where the OMLU control cards go within a translator job stream. The parameters shown convert all the modules on SYSUT1 to an OML for subsequent transcription to disc by DOMLMR.

Once a module is transcribed to disc, it will remain there until it is deleted via a console message. The program will not request modules for deletion when run under monitor.

When a given module is processed through DOMLMR, the module directory on disc is searched to see if the module already exists. If it does, it is replaced by the new version. If it does not exist, it is added to the file. The only restriction to the use of the DOMLMR is that MODULES CONTAINING A SORT MAY BE TRANSCRIBED TO DISC, BUT THE LINKAGE EDITOR WILL NOT BE ABLE TO LINK THEM BACK. If you do put a sort module out to the library, inform operations so that they may delete it to free up the disc space.

```
// STARTM
// JOB
// PARAM
// translator (ASSMBL, COBOL, FORTRN, RPG)

     SOURCE DECK (MODA)

// translator

     SOURCE DECK (MODB)

// EXEC OMLU
 COPY NONE
 CATALO SYSUT1
 END
// EXEC DOMLMR
// LNKEDT
 PROG MODAB
 INCLUDE SYSOML(MODA,MODB)
// ENDMON
```

MODA and MODB will be cataloged as separate modules. If MODA or MODB or both must be re-translated at a later date, the'// EXEC OMLU' thru '// ENDMON' control cards remain the same. For example, if MODB required re-translation, it would be transcribed to disc, following OMLU, and then linked with the original version of MODA.

MAINTENANCE DOCUMENTATION FOR DOMLMR

The logic used to reformat a given record type (00-05) on the input tape OML for the disc resident OML was determined by manual R/A edit comparison after CLTR to a tape edit of the input OML. The reformat logic as determined by this process is explained in appendix A. The same method is quite reliable for determining a problem which may be uncovered in DOMLMR today. As of the time this document is being written, DOMLMR is maintaining over 200 modules on a 70/590 without a problem.

I recommend that prior to any change to DOMLMR, you thoroughly familiarize yourself with this program via the documentation provided. A brief description of each of the sub processing logic modules used is provided along with a detail logic chart. Appendix A and B should also be reviewed prior to going through the main line code of the detail logic charts.

The GETEM macro, which prints out at the front of the assembly listing, is used for random access device assignment to eliminate the need for run time parameters. The function of this macro is to build an extent matrix for the VSN and file name indicated at, in this case, tag NAME3.

The only program change required to implement this utility is the VSN number located at tag NAME3. The program is preset for 70/564 use and modifies itself for 70/590 use if that is the device assigned.

The main line code of the program has notes indicated on the logic chart by double circles which are keyed to the notes themselves. The Sub Processing Notes section is keyed to sections by name and a more explanatory logic chart has been drawn. The Sub Processing Notes are designed for handy reference during your analysis of the main line code to explain the action taken by the 'BAL' blocks.

PREOML PROGRAM NARRATIVE

The PREOML utility is used to preformat the disc extent used with the DOMLMR program. This utility will preformat a 70/590 or 70/564 disc. The GETEM macro is used for device assignment. The volume serial number of the volume to be initialized is at tag NAME 3 card number 02990. The following operations are performed by this program:

1. Verify that the extent starts on cyl 1 track zero and is allocated in full cylinder increments. If the program is being run to extend an existing area, the new ending cylinder number must be greater than the old ending cylinder number.
2. Write 1688 or 1129 byte records of all hex zero to every track in the extent (4 records of 1688 bytes per track for 70/590 or 3 records of 1129 bytes per track for 70/564).
3. The first record of the Directory is formatted as indicated in appendix A. All records in the Directory are chained together by position 1-5. The records in cyl 2 through n are not chained.
4. The control record is contrructed and written to track nine cylinder one. The cylinders that are allocated are set to hex zero in the track table of the control record.

The portion of the track table representing area beyond the allocated area is set to hex '77' for 70/564 or hex 'FF' for 70/590 to indicate full.

5. When PREOML is run to extend on existing extent, the newly acquired area is preformatted with dummy records and the control record is updated to reflect the new extent limits.

SUB PROCESSING NOTES

SECTION NAME: MOVE LOGIC ENTRY POINT: MOVE

FUNCTION: This logic will move the number of bytes specified by the half word constant following the BAL instruction from the address contained in GR 4 to the address contained in GR 5. If the field to be moved is larger than the remaining bytes in the output area (GR 6), the field moved is split to fill up the current output block, that block is written to disc, and the remainder of the field is moved to the start of the next block. After the move is complete, GR 4 and GR 5 are incremented by the number of bytes moved.

SECTION NAME: WRITE LOGIC ENTRY POINT: WRITE 1
CLEAR
CLEAROT

FUNCTION: This logic will write the block contained in OTARA to disc. It uses the Table To Address (TTOA) logic to determine the 'write to' address on disc. The address (CCHHR) of the next logical record is placed in position 1-5 of the current output record unless the switch at tag OTSW is turned on. When the record is written, GR 6 is set to indicate a full block and GR 5 is set to OTARA+8 in preparation for the next move. The

first disc 'write to' address is generated by the TTOA logic called in the 01 processing for initialization. This routine is entered at tag CLEAR in the housekeeping logic and tag CLEAROT in the 01 logic.

SECTION NAME: CALCULATE NEXT AVAILABLE CCHHR FROM TABLE — ENTRY POINT: TTOA

FUNCTION: This logic generates a CCHHR address for disc based on the next available address in the track table portion of the control record. The generated CCHHR address is placed at tag DCCHH. If the extent is exhausted, it is detected in this logic. The number of remaining data records counter in the control record is decreased by 1.

SECTION NAME: RESET USED BIT IN TABLE BASED ON CCHHR ADDRESS — ENTRY POINT: ATOT

FUNCTION: This logic sets the bit in the track table to off that corresponds to the CCHHR address at tag 'DCCHH'. The number of remaining data records counter in the control record is increased by 1.

SECTION NAME: READ LOGIC FOR RECORD TYPE 00 — ENTRY POINT: READO
READ LOGIC FOR RECORD TYPE 01 — ENTRY POINT: READ1

FUNCTION: The disc record specified by CCHHR in tag SEKADR+3 is accessed in OTARAOO for a 00 record or OTARA for a 01

record. The address of the record just accessed (CCHHR) is stored for subsequent use by the REWRITE logic for 00 and 01 records. The CCHHR address in tag SEKADR+3 must be provided prior to entering the READ logic.

SECTION NAME: USED TO REWRITE UPDATED 00 RECORD — ENTRY POINT: WRITE0

USED TO REWRITE UPDATED 01 RECORD — ENTRY POINT: WRITE01

FUNCTION: The record currently in core at tag OTARA00 for the 00 record or OTARA for 01 record is written to disc at the address stored by a previous read (READ0 or READ1).

SECTION NAME: LOCATE MODULE NAME IN 00 RECORD — ENTRY POINT: DLØØ DLØØD

FUNCTION: The 8 byte module name at tag DELPRG is located in the directory. If the module name is not in the directory, the address returned points to the next sequential directory entry. The address of the left end of the directory entry in OTARAØØ is returned in GR 1. A direct branch to tag DLØØD is used in the print logic along with setting the switch at tag 'PONLY' for the serial access of the directory.

SECTION NAME: LOCATE 01 RECORD BASED ON 00 POINTER — ENTRY POINT: CHASEØ1

FUNCTION: The CCHHR address of the 01 record for the 00 record indicated by GR 1 is accessed into tag OTARA. GR 1 is set to the CCHHR address in the 01 record which points to the first data record for the module. (See appendix A). This module is designed to be used following the DLØØ logic.

SECTION NAME: CHASE DATA RECORDS FROM 01 TRANSACTION ENTRY POINT: CHASEØ2

FUNCTION: Reset the corresponding bit in the track table to OFF for the 01 record just accessed and read and reset all corresponding bits for all 02 through 05 data blocks for the module. The last block for module will contain hex zero in pos 1 through 5.

SECTION NAME: SHIFT 00 RECORD TO DELETE ENTRY ENTRY POINT: SHIFTØØ

This logic shifts the 00 record at tag OTARAØØ 16 positions left to eliminate an entry. GR 1 must point to the left end of the module to be deleted (set up by DLØØ logic).

SECTION NAME: MODULE NAME PRINT ROUTINE ENTRY POINT: PRINTO

FUNCTION: This logic prints a listing of all the module names contained in the directory. It is used in the deletion logic as a result of the $ PRINT operand.

NOTES FOR MAIN PATH LOGIC

Note	Chart Number	HOUSEKEEPING SECTION
1	1	The CLEAR logic is located in the WRITE logic section. The OTARAOO and OTARA are cleared to hex zero. The return address from the CLEAR logic is a preset initial value of HSKP+5 at tag ST14 which is loaded into GR14 at the end of the WRITE logic.
2	1	This is done to prevent problems in the end of job logic in the event of a null input tape.
3	1	All reads and writes to disc will BAL 14 to tag RTRY in the event of an I/O error. This tag is located in the GETEM macro. The GETEM macro maintains a count of the number of re-tries and types a message if the count is exhausted.
4	1	The 'R' in the word 'DOMLMR' located in the control record (see appendix B) is used for a file lock/unlock indicator. If the program terminates abnormally, the file will be locked. This prevents DOMLMR from changing data in the file until the maintenance programmer can analyze the problem. The indicator is reset by re-establishing the file with DDRL.
5	1	Control record verification is made by a compare of the constant 'DOMLMR' to the first six positions of the control record read. See appendix B for the format of the control record.
6	1	If the ending CCHH in the extent matrix does not match the CCHH in the control record, the program terminates. Probably caused by the size of the extent being extended by RAALR and not running PREOML to update the control record and preformat the new area prior to running DOMLMR.
7	1	The 'R' in the word 'DOMLMR' is reset at this point. When the control record is written back to disc at normal end of job, it will unlock the file.

8	2	The date in the executive is compared to the date in the first record in the Directory. See Directory Record format in appendix A.
9	2	Because the module names entered via the console can be variable length, they must be expanded to eight positions prior to searching for them in the Directory.
10	3	The comparison of tag DELPRG to the address in GR 1 indicates if the module was found in the directory or not. (See DLØØ logic.)
11	3	The letters 'CCHHR' are placed in the Directory record in place of an actual disc address until the '01' (Index Record) is read from the input tape and on actual disc address is calculated for the 01. This compare was mainly used for debugging.
12	3	Setting DELPRG to hex zero causes the DLØØ logic to access the first directory block and point to the first directory record following the dummy record (see appendix A).
13	3	Setting the NOP switch in the DLØØ logic causes serial access of every record of every block in the directory.
14	3	PRINTO, the logic module that formats and prints the directory listing, uses logical level FCP. For this reason, the address generated by the DLØØ logic in GR 1 must be shifted to GR 5 to be saved.
15	3	The high dummy Directory entry (see appendix A) signals end of job to the print logic module (PRINTO) and breaks the print loop.
16	3	Tag DLØØD is located in the DLØØ logic module. Because the DLØØ logic was last entered by a BAL 14 (just prior to tag PLISTB) it will continue to return to that tag until it is reset by a BAL 14 from another location.

17	4	The DTFSR for the input tape is set for unlabeled tape. The 3 EXCPW macros rewind the tape and position it in front of the first tape mark for the open macro. The DTFSR is also set for no rewind at open.
18	4	The first data record following the tape mark on a tape OML is a special identifying record.
Note	Chart Number	OBJECT MODULE DIRECTORY BLOCK (00) PROCESSING
19	6	General registers 5, 6 and 7 are loaded from tag RECØØ. GR 5 contains the starting address of OTARAOO+8 in the RECØØ area. This value is used as the 'TO' address in the MOVE logic. After a move, GR 5 is incremented by the length of the move to set up for the next move. GR 6 contains the number of bytes that will fit in the current record. This register is decremented by the number of bytes moved. GR 7 points to tag RECØØ. This is used to access reset values for the first two entries (GR 5 and 6) and to access the storage area for the CCHHR of the 00 record currently in core. (See READØ and WRITEØ logic.)
20	6	The MOVE logic will automatically write a record when the I/O area is full as determined by the remaining byte count in GR 6. This logic is not desirable for the 00 record processing. By loading GR 6 with the value of 2000, this feature is not used.
21	6	The NOP at tag MEXØ is set to go to tag EMØ.
22	6	The 00 record contains multiple module names and is fixed length. The unused portion of the tape record following the last module name is filled with hex zero. In the event that the 00 record is full, GR 10 is loaded with the address of the last input record position plus 1. As GR 4 is moved from one module to the next, it is compared to GR 10 for equality.
23	6	This switch is turned on at 'A' page 7 when a new module is to be added to the file. When the switch is 'on', the 00 record is rewritten to disc.

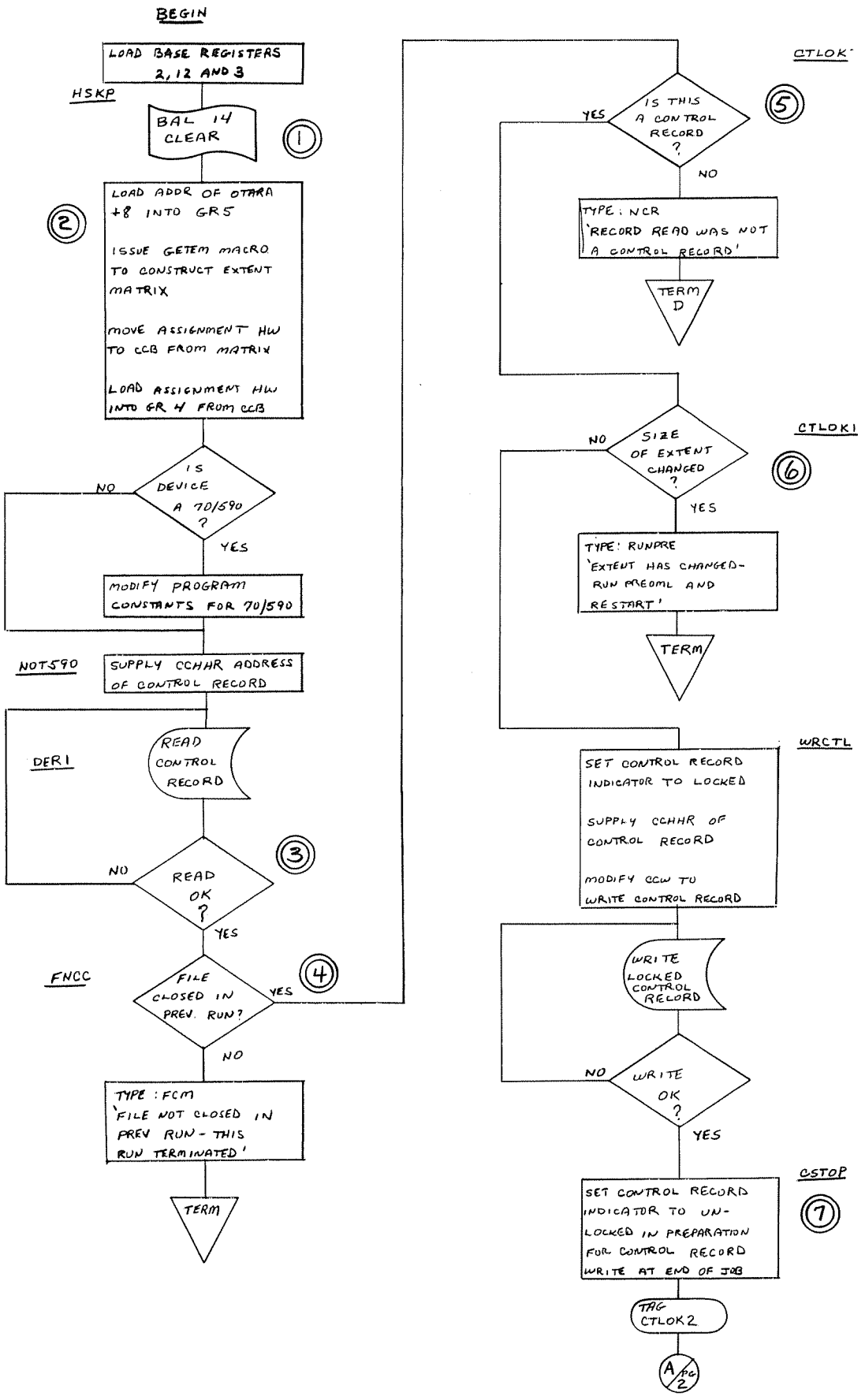
24	6	This switch is turned on when an entry was shifted out of the block to make room for an addition. (See tag OFØØ page 7.)
25	6	Go read the 01 record in the data position of the extent into OTARA.
26	6	Go access all records for the module in the data position of the file and set the bits in the track table to not used.
27	7	This block saves the current address of the module being processed in the tape input area. It then sets the 'TO' address for the move logic (GR 5) to the end of the OTARAOO area +1 and the 'FROM' address for the move logic to the last 16 byte entry in in the OTAROO (70th entry for 564, or 105th entry for 590). The BAL to 'MOVE' at tag MAGN moves 16 bytes and increments GR 4 and GR 5 the length of the move.
28	7	The switch at MEXØ is turned on when the last move is about to be made. The normal ON for this switch goes to tag EMO. The switch is altered to go to tag OFØØB at tag ONFR on page 8. This setting is used if more than one directory block has to be accessed and shifted to make room for a module entry.
29	7	GR 1 contains the address in the OTARAØØ block of where the new module should be merged. A compare of GR 4 (from address of move) to GR 1 for equality determines the last module is about to be moved and sets the switch at MEXØ to break the loop. If GR 4 valve is less than GR 1, the error branch tag TERMDA is taken.
30	7	If the error condition at note 29 is sensed, it is not an error if the module to be added belongs in the last position of the record.
31	7	Record shifted out is determined by a compare of hex zero to the end of OTARAOO+1. If the area

		contains hex zero, no record has been shifted out. If a record has been shifted out, it is put in the first entry of the next 00 block.
32	8	GR 1 is set to the address of OTARAØØ+8. This causes the logic at tag SHIFTØØ to move every entry in the 00 record 16 bytes right to free up the first entry for the entry shifted out the end of the previous record.

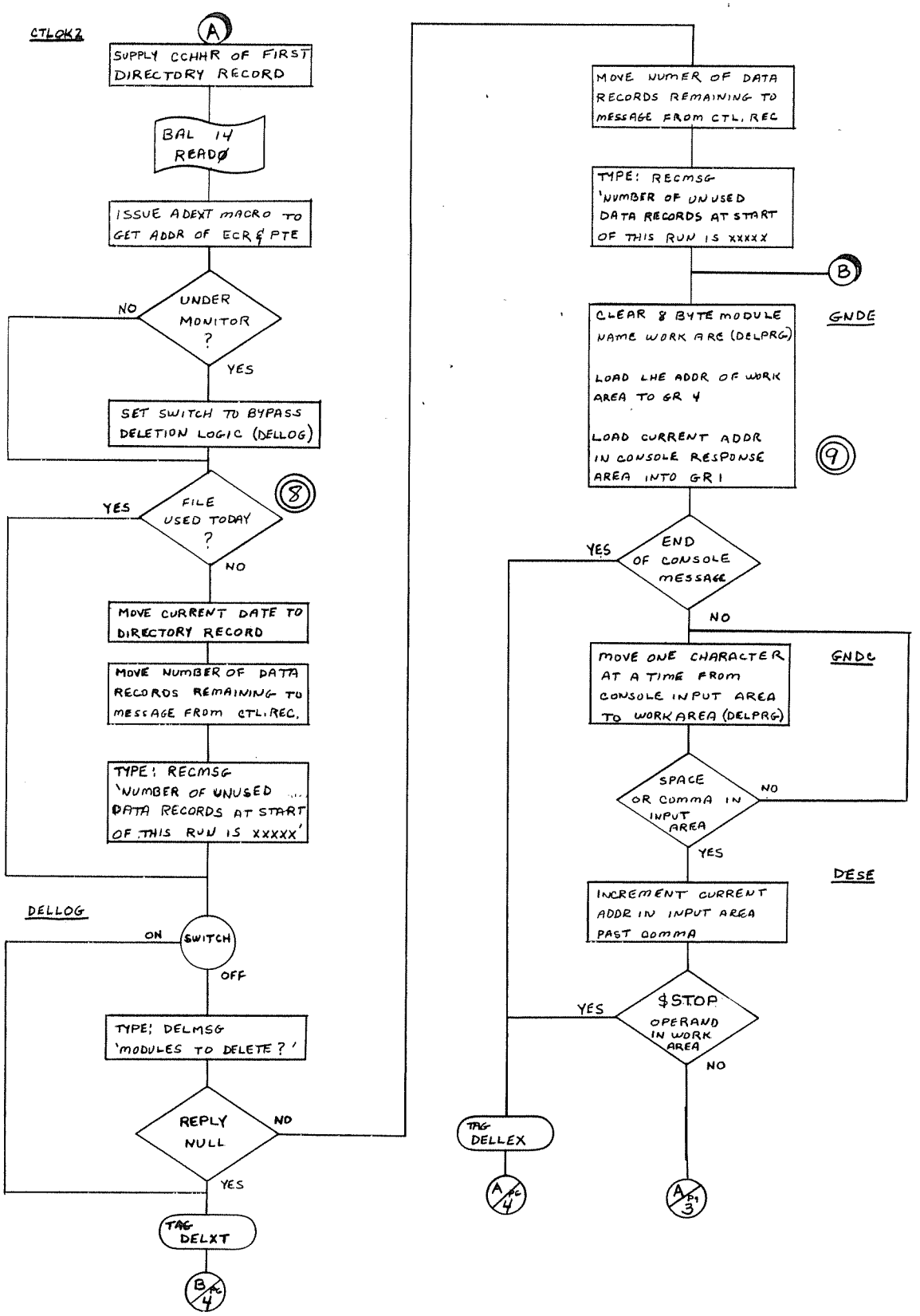
Note	Chart Number	INDEX BLOCK (01) PROCESSING
33	9	If multiple 01 blocks are read for a single module, they all contain the same indicators for the number of entries, externs and common as well as the same constant info. For that reason, this branch goes directly to the logic to move the 'N' fields contained in the second and following records. (See appendix A-4)
34	9	This is the first call in the program for an address generation from the table. The TTOA logic generates the CCHHR of the first available disc address at tag DCCHH. The WRITE logic keeps track of the next available disc address to put in bytes 1 thru 5 of the current record by issueing all subsequent calls for logic module TTOA.
35	9	The area OTARA is cleared to hex zero initially at this point. The WRITE logic clears this area after each WRITE.

Note	Chart Number	INDEX BLOCK 01 PROCESSING
36	10	The 'move' logic will write the 01 record to disc if the output area is full. In this case pos 1-5 of the first 01 record will contain the CCHHR of the next 01 record.
37	10	See the Utilities Manual for definition of SUB CODE in position 2 of the input tape for a 01 type record.
38	10	Since 01 records are written under control of the WRITE logic used with the MOVE logic, the CCHHR of the next record is placed in pos 1-5 of the record. Control is transferred to this block when the last 01 record for the module has been constructed. Since this is the last data record for this module until the 02 record is later accessed from tape, it is so indicated by the hex zero in pos 1-5. Setting the switch in the WRITE logic inhibits the moves of CCHHR to overlay the hex zero.

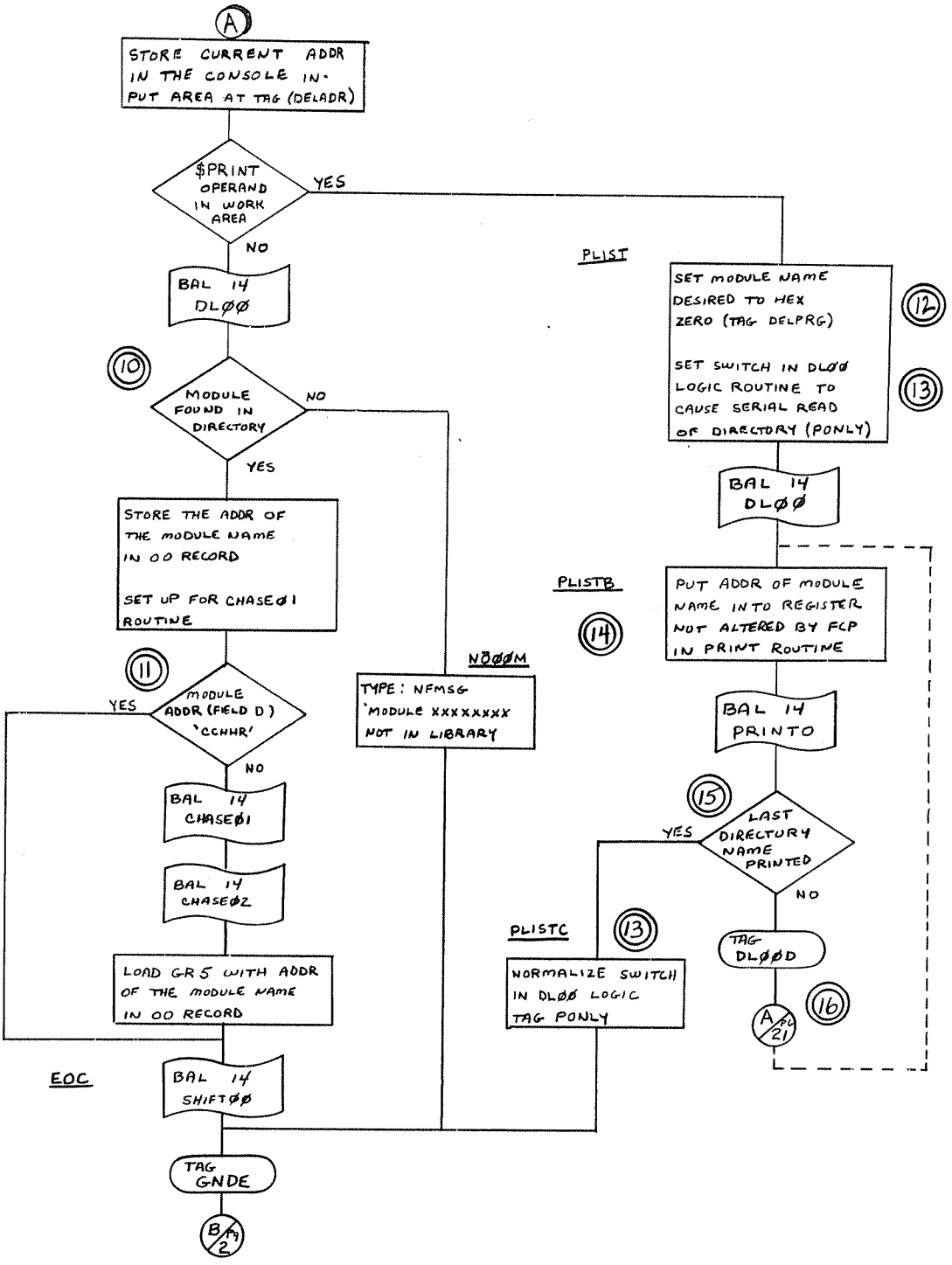
Note	Chart Number	OBJECT MODULE DESCRIPTOR BLOCK (02)
39	11	The 02 records on tape represent the first record of a new module. The first 02 accessed will follow the last 01 record on tape and will cause the switch at tag D02 to be turned off. When the switch is off and an 02 record is read from tape, the last record for the previous module is written to disc with positions 1-5 of that record set to hex zero as an end of module indicator. The number of used bytes in the record is also calculated and put into the record.
Note	Chart Number	EXTRN BLOCK PROCESSING (03)
40	12	This move takes all the extrns in one move to the output record. GR 4 and 5 are set to the correct addresses by the previous move. The number of bytes to move is calculated in the logic at tag D03B.
Note	Chart Number	TEXT BLOCK (04) AND MODIFIER BLOCK (05) PROCESSING
41	13	When the address of tag LEN04 is sent to the move logic in Gr 4, the move logic will check to insure that there is at least 10 bytes remaining in the output area prior to the move. If less then 10 bytes remain, the current record is written to disc and the data in tag LEN04 is put at the front of the next record. This is done to insure that fields A thru D are not split over two records.
Note	Chart Number	END OF JOB PROCESSING
42	14	Since the WRITE logic maintains the CCHHR of the next available disc address, it must be reset to unused at end of job. The table entry was initially set to used when the CCHHR was generated in the ATOT logic.



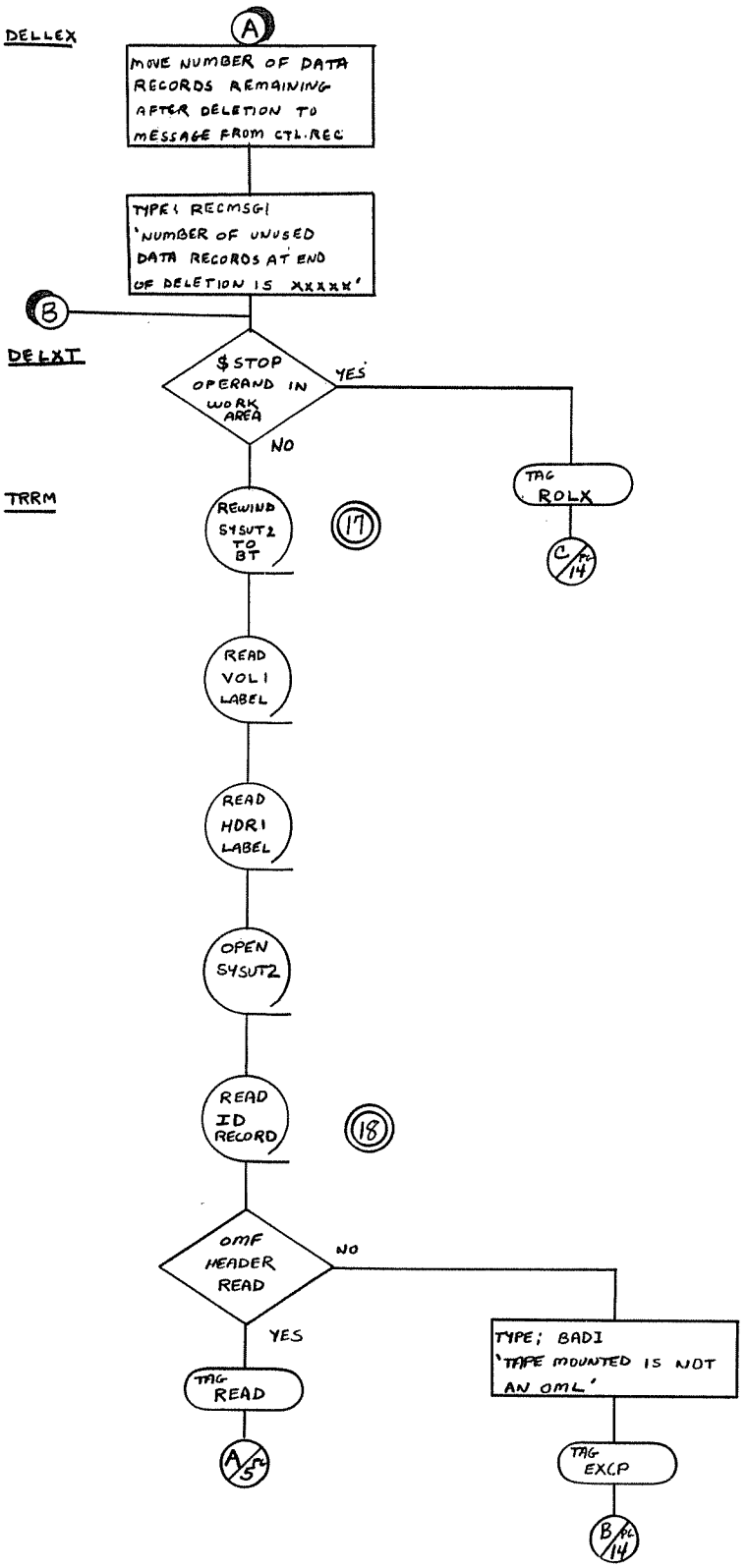
RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title: HOUSEKEEPING LOGIC	Date	Chart No.: 1 of 24
System Title: DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	Program Title: ACCESS AND VERIFY CONTROL RECORD	Revision: Letter	Date



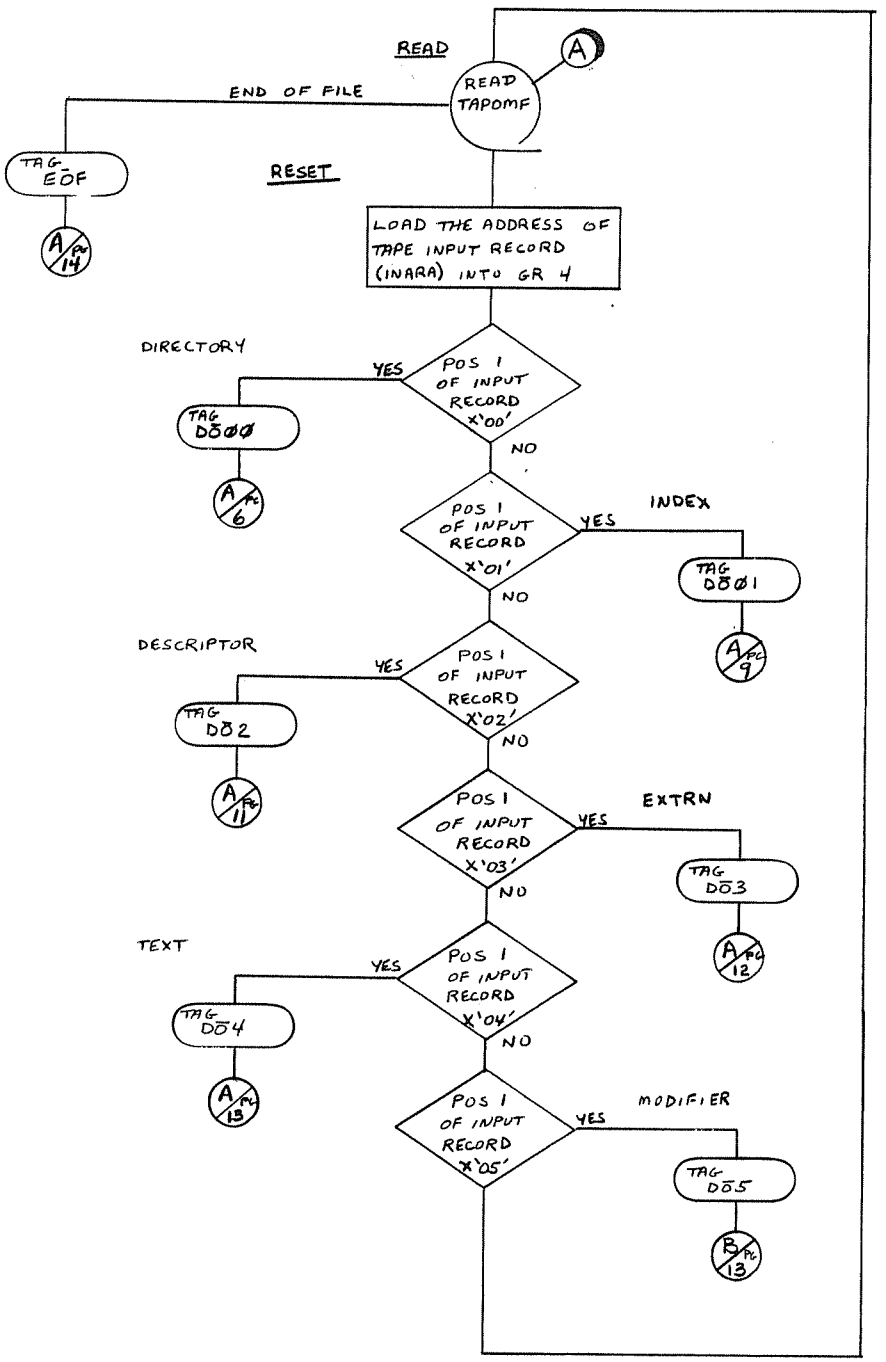
RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title: HOUSEKEEPING LOGIC	Date	Chart No. 2 of 24
System Title: DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	Program Title: DELETION LOGIC	Revision: Letter	Date



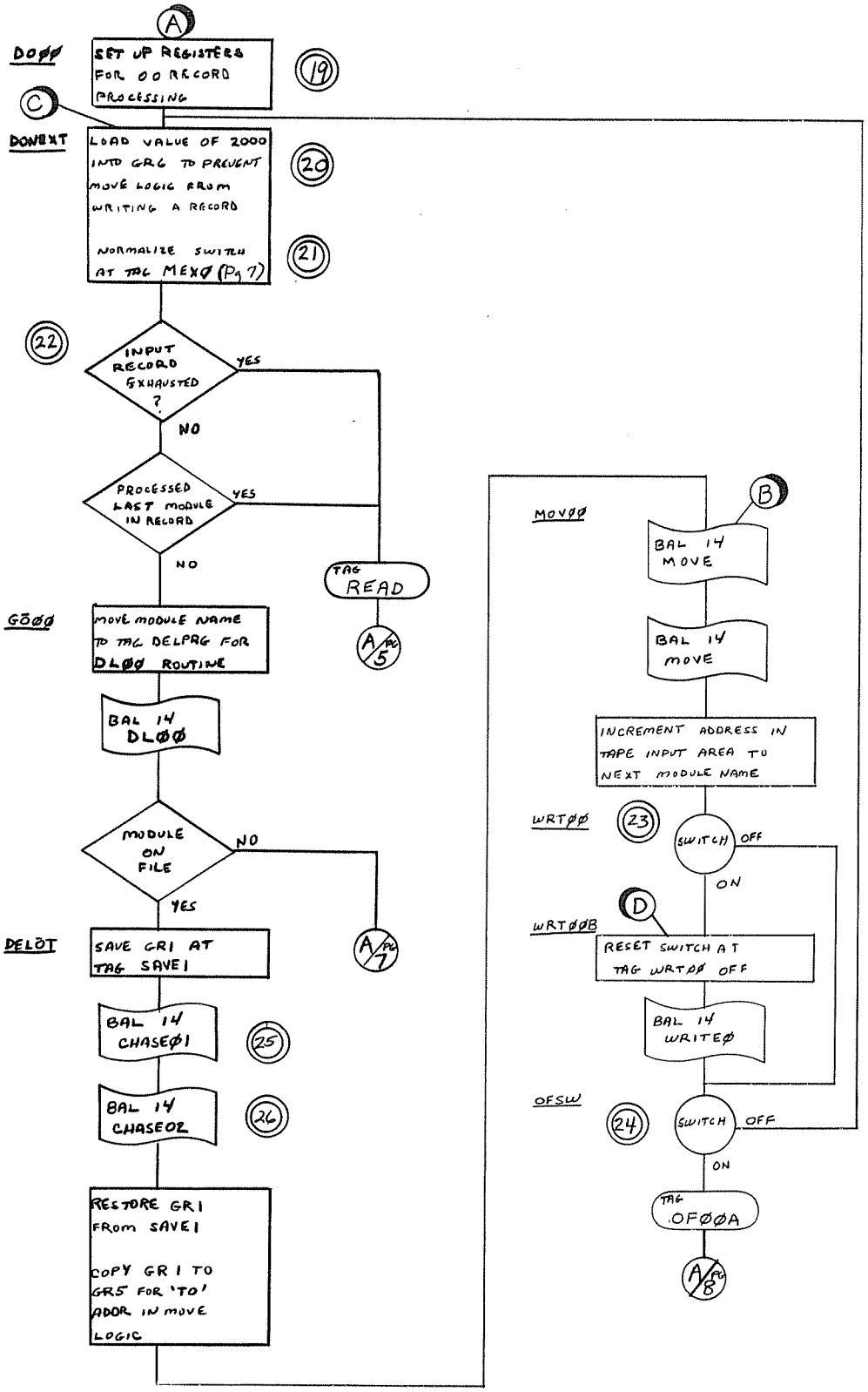
RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title: HOUSEKEEPING LOGIC	Date	Chart No. 3 of 24
System Title: DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	Program Title: DELETION LOGIC	Revision: Letter	Date



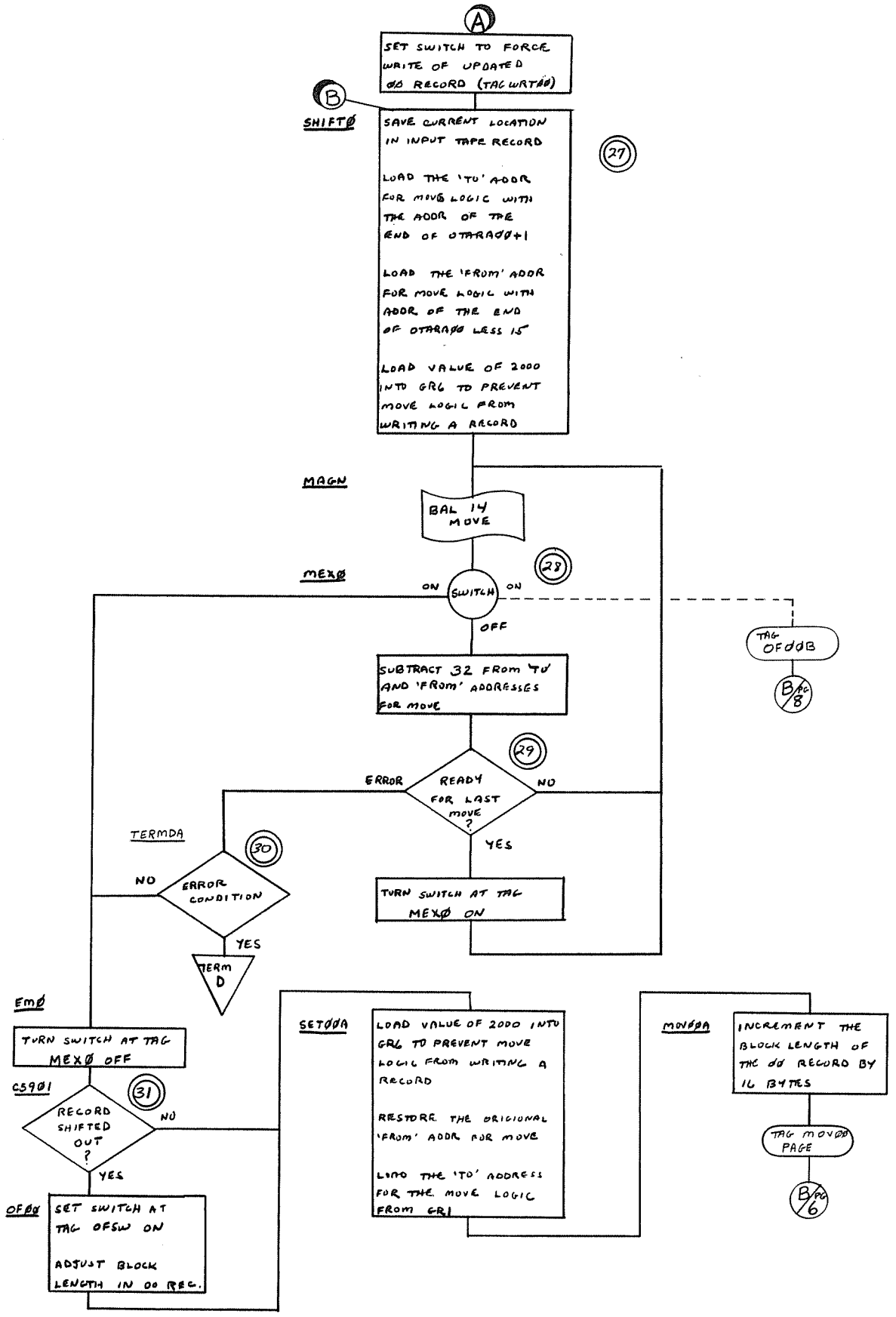
RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title: HOUSEKEEPING LOGIC	Date	Chart No. 4 of 24
System Title: DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	Program Title: DELETION LOGIC EXIT & INPUT TAPE VERIFICATION	Revision: Letter	Date



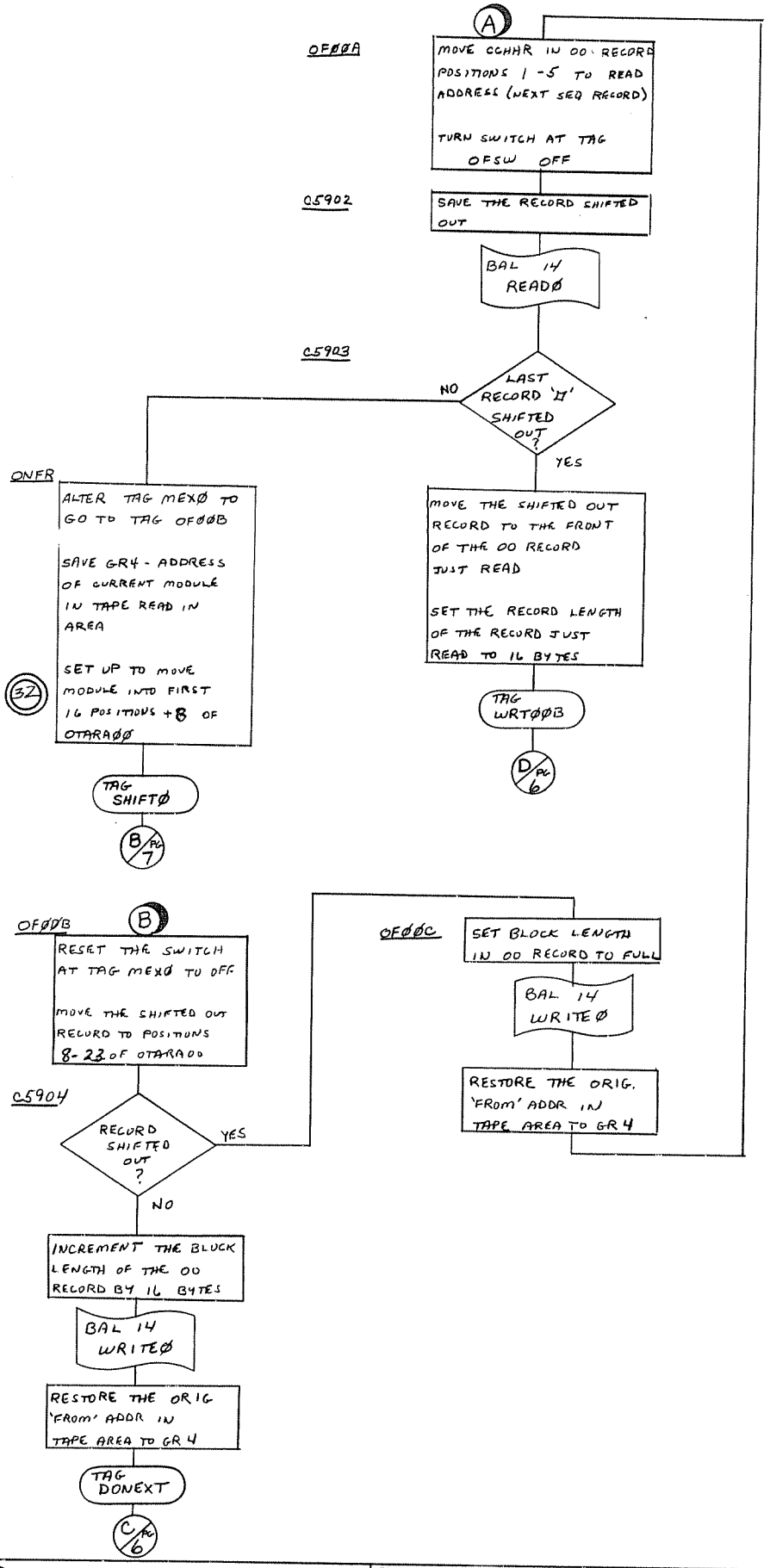
RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title: TAPE READ AND INPUT RECORD TYPE DETERMINATION LOGIC	Date	Chart No. 5 of 24
System Title: DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	Program Title	Revision: Letter	Date



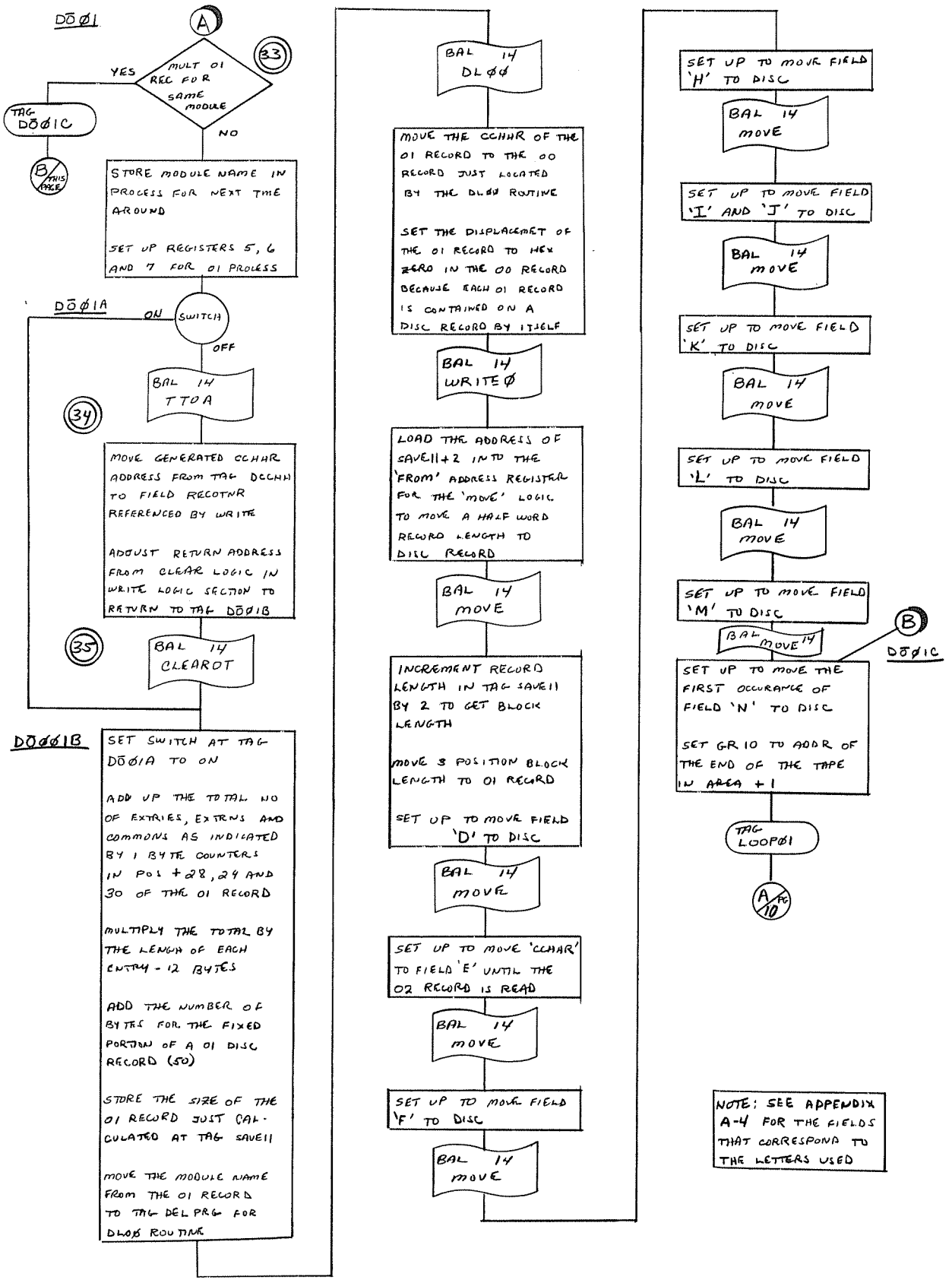
RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title: OBJECT MODULE DIRECTORY BLOCK (00) PROCESSING	Date	Chart No. 6 of 24
System Title: DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	Program Title: MAIN LINE CODE	Revision: Letter	Date



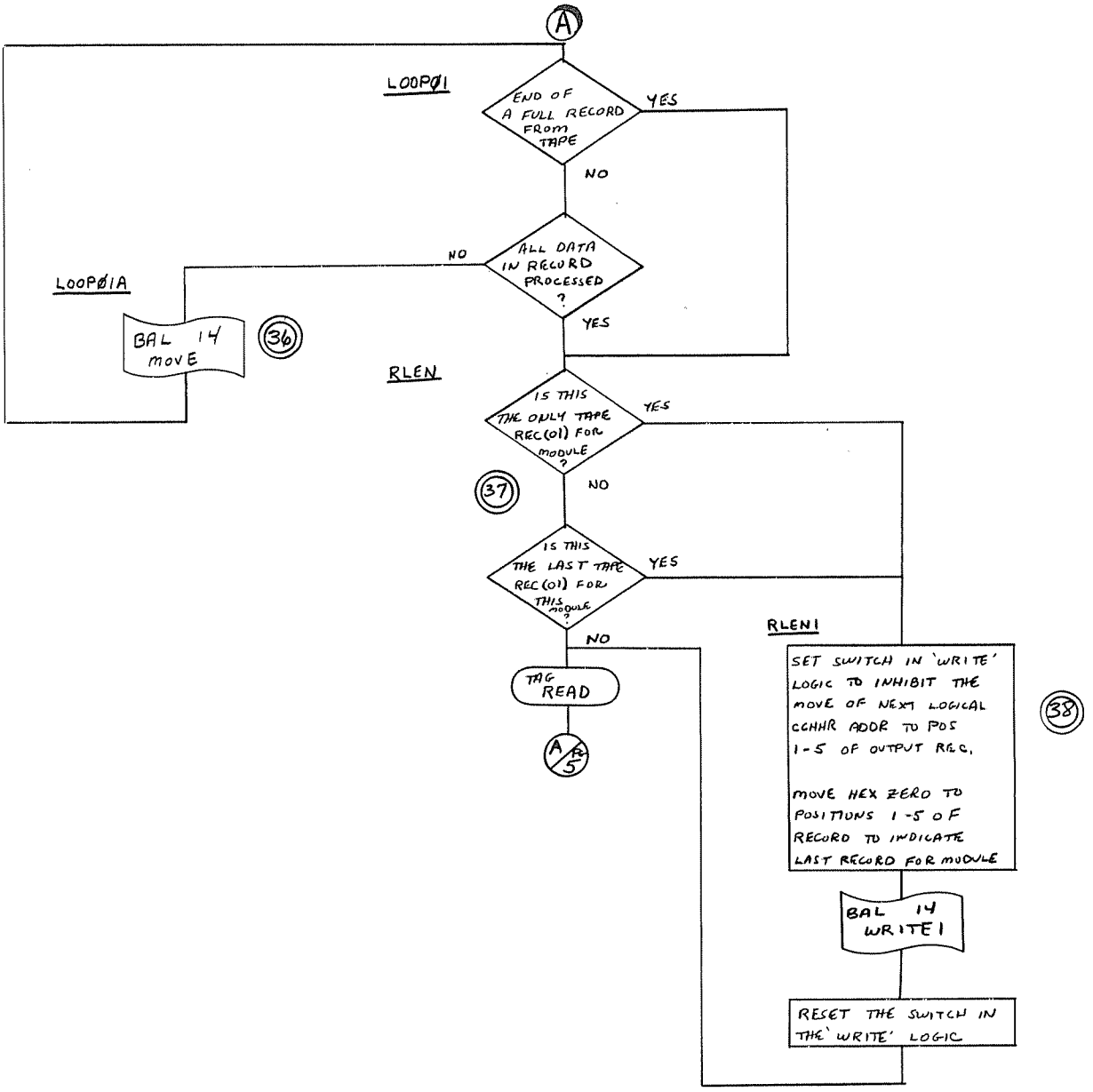
RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title: OBJECT MODULE DIRECTORY BLOCK (00) PROCESSING	Date	Chart No. 7 of 24
System Title: DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	Program Title: LOGIC TO SHIFT THE 00 RECORD TO MAKE ROOM FOR AN ADDITION TO THE FILE	Revision: Letter	Date



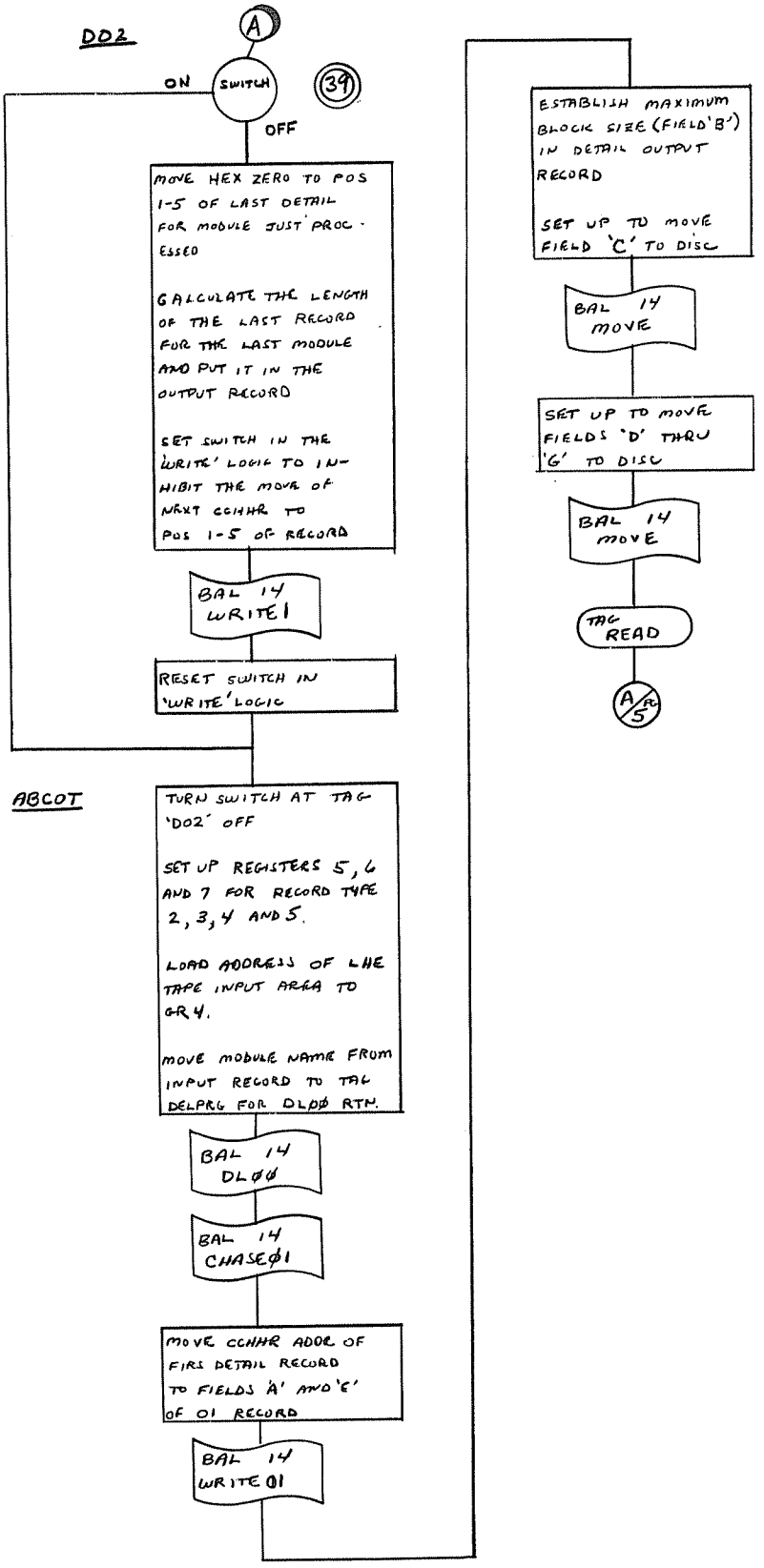
RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title: OBJECT MODULE DIRECTORY BLOCK (00) PROCESSING	Date	Chart No. 8 of 24
System Title: DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	Program Title: LOGIC TO PROCESS TRUNKATED RECORDS AS A RESULT OF A SHIFT	Revision: Letter	Date



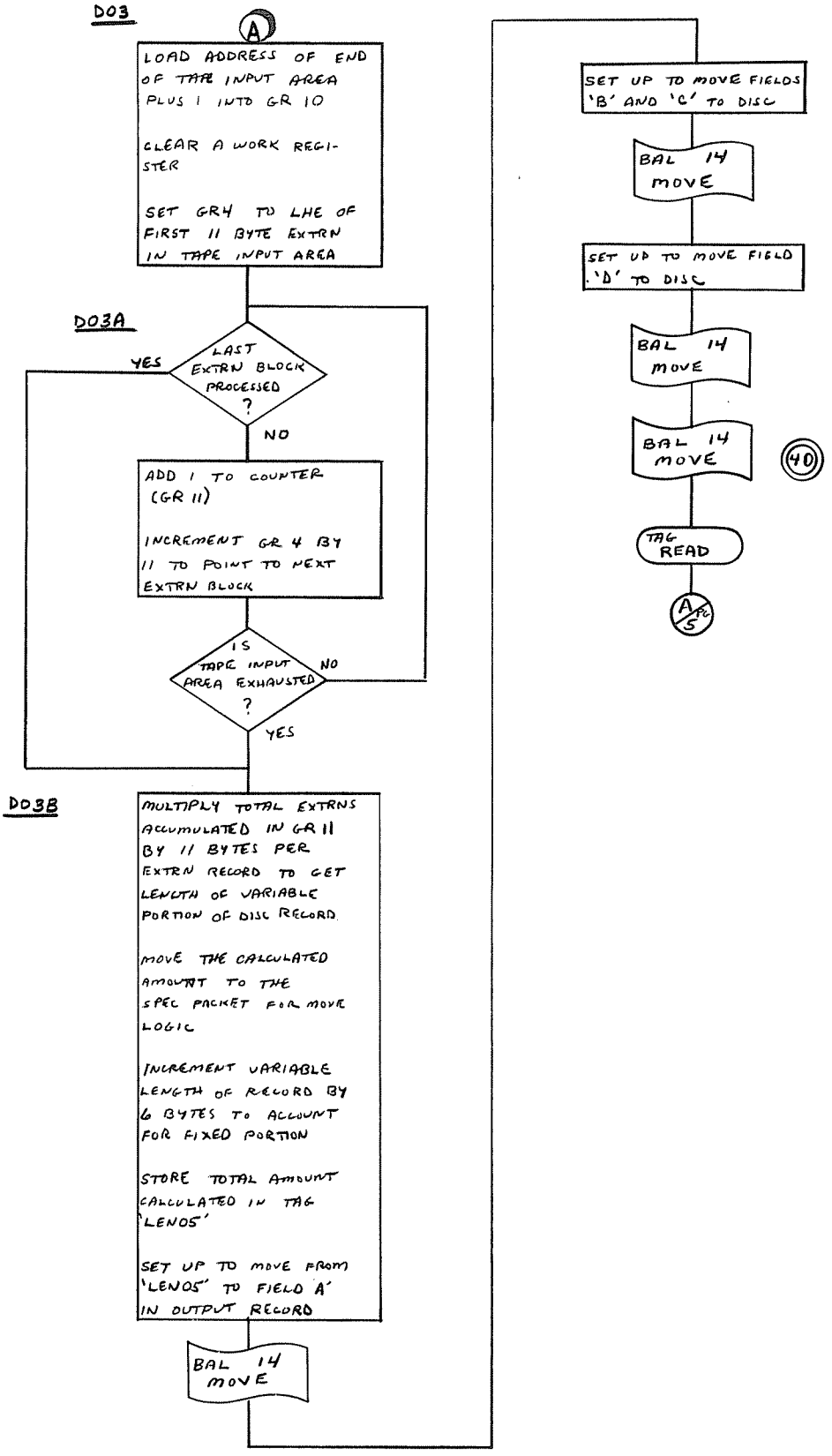
RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title: INDEX BLOCK (01) PROCESSING	Date	Chart No. 9 of 24
System Title: DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	Program Title	Revision: Letter	Date



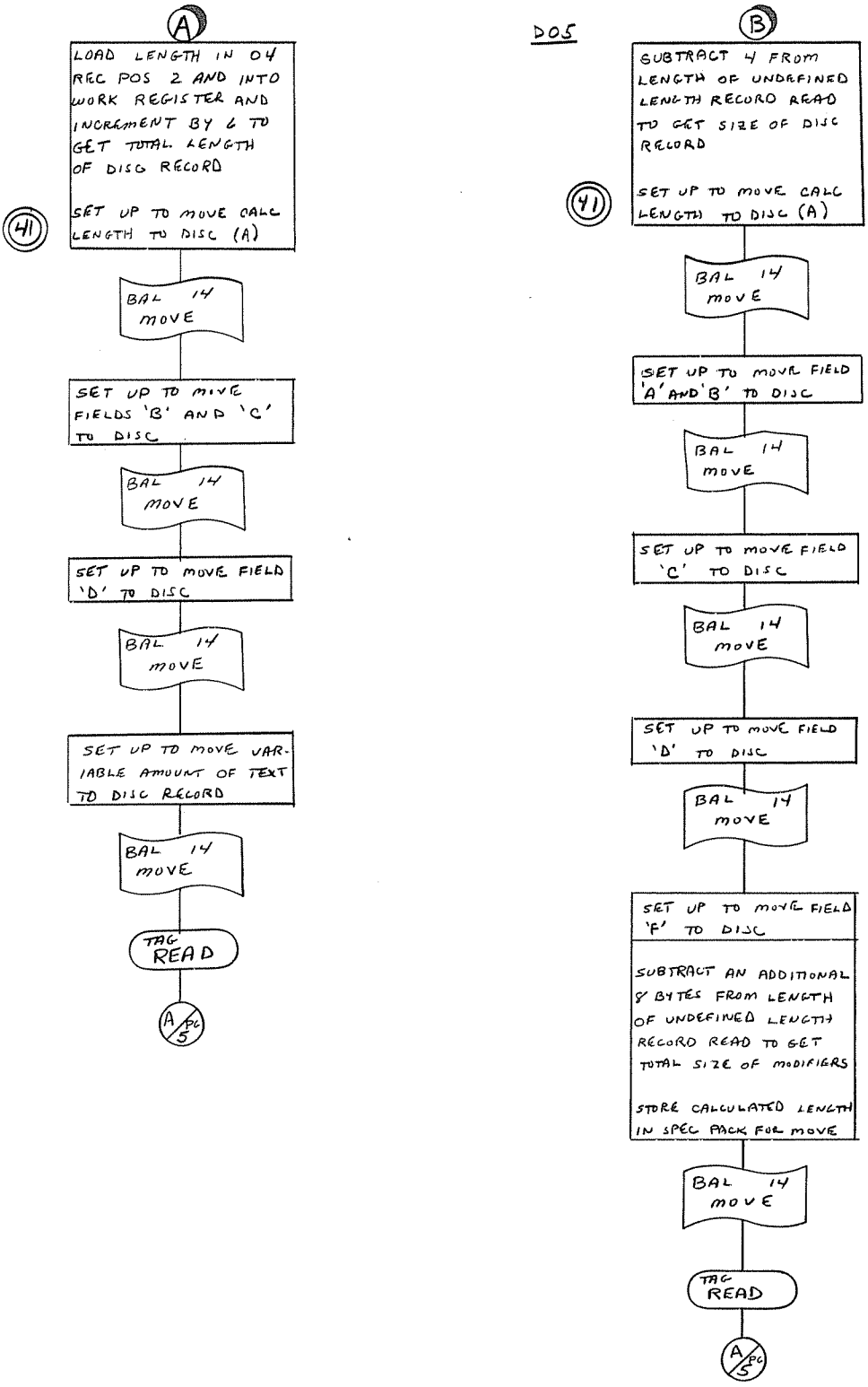
RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title: INDEX BLOCK (01) PROCESSING	Date	Chart No. 10 of 24
System Title: DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	Program Title	Revision: Letter	Date



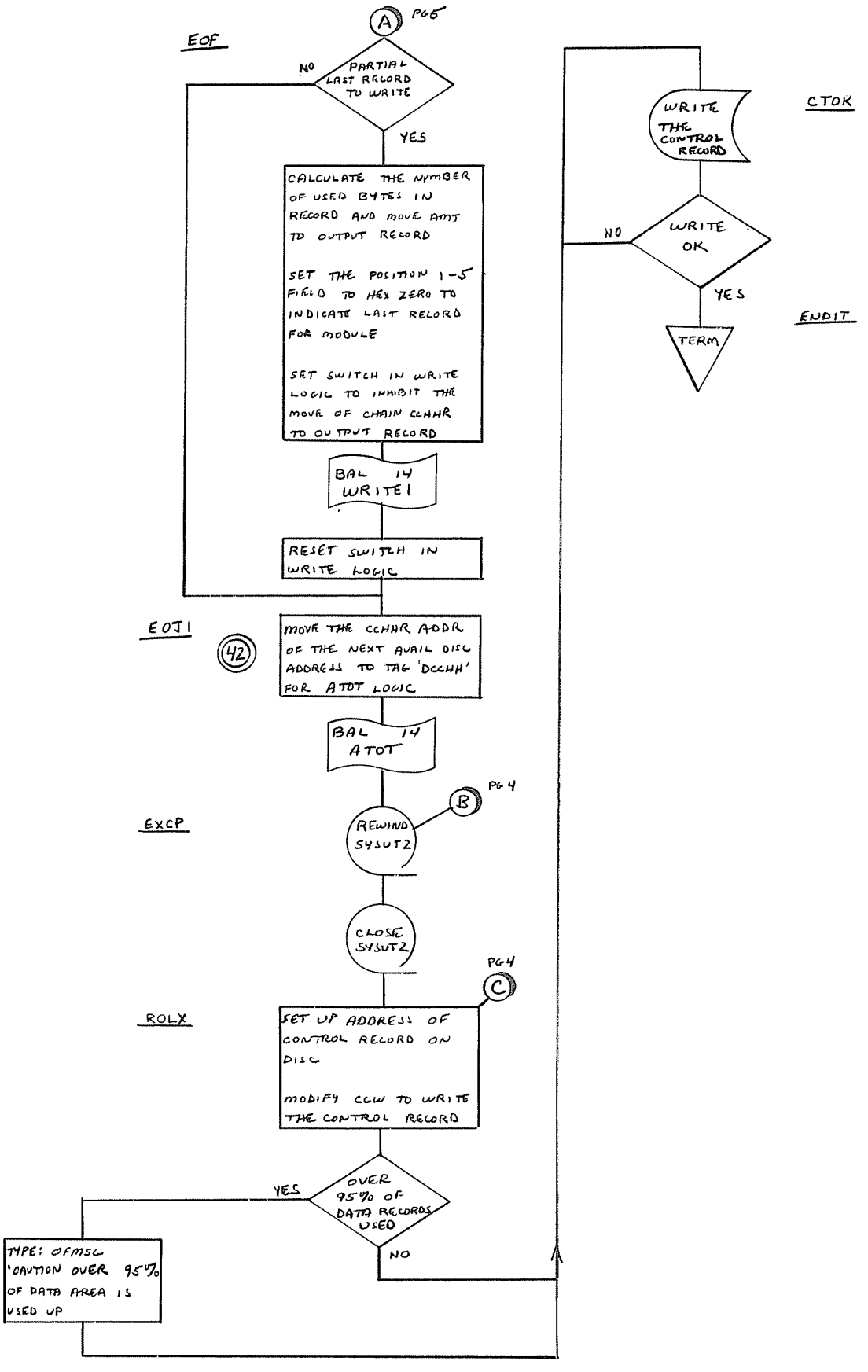
RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title: OBJECT MODULE DESCRIPTOR BLOCK (02) PROCESSING	Date	Chart No. 11 of 24
System Title: DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	Program Title	Revision: Letter	Date



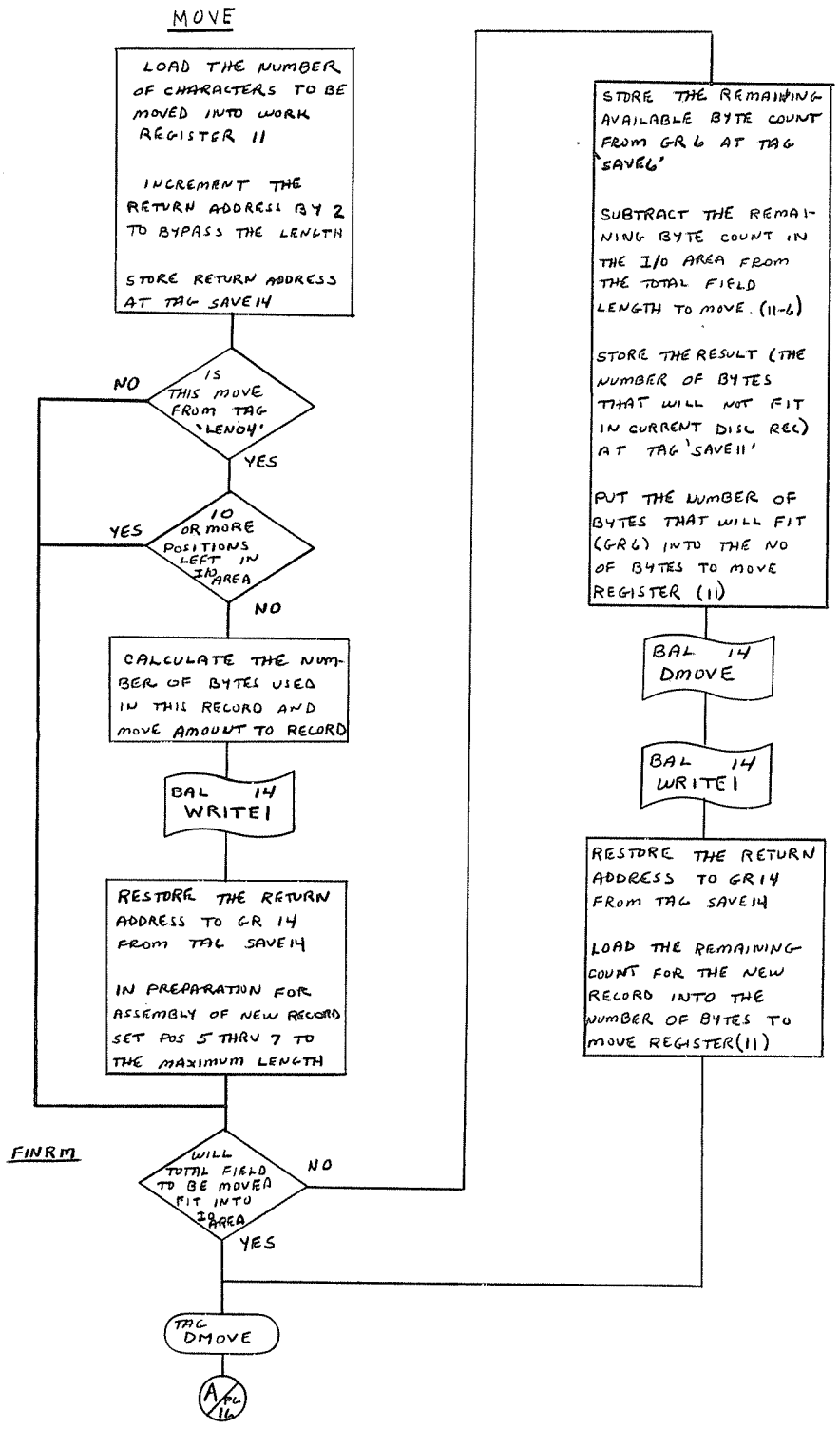
RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title: EXTRN BLOCK (03) PROCESSING	Date	Chart No. 12 of 24
System Title: DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	Program Title	Revision: Letter	Date



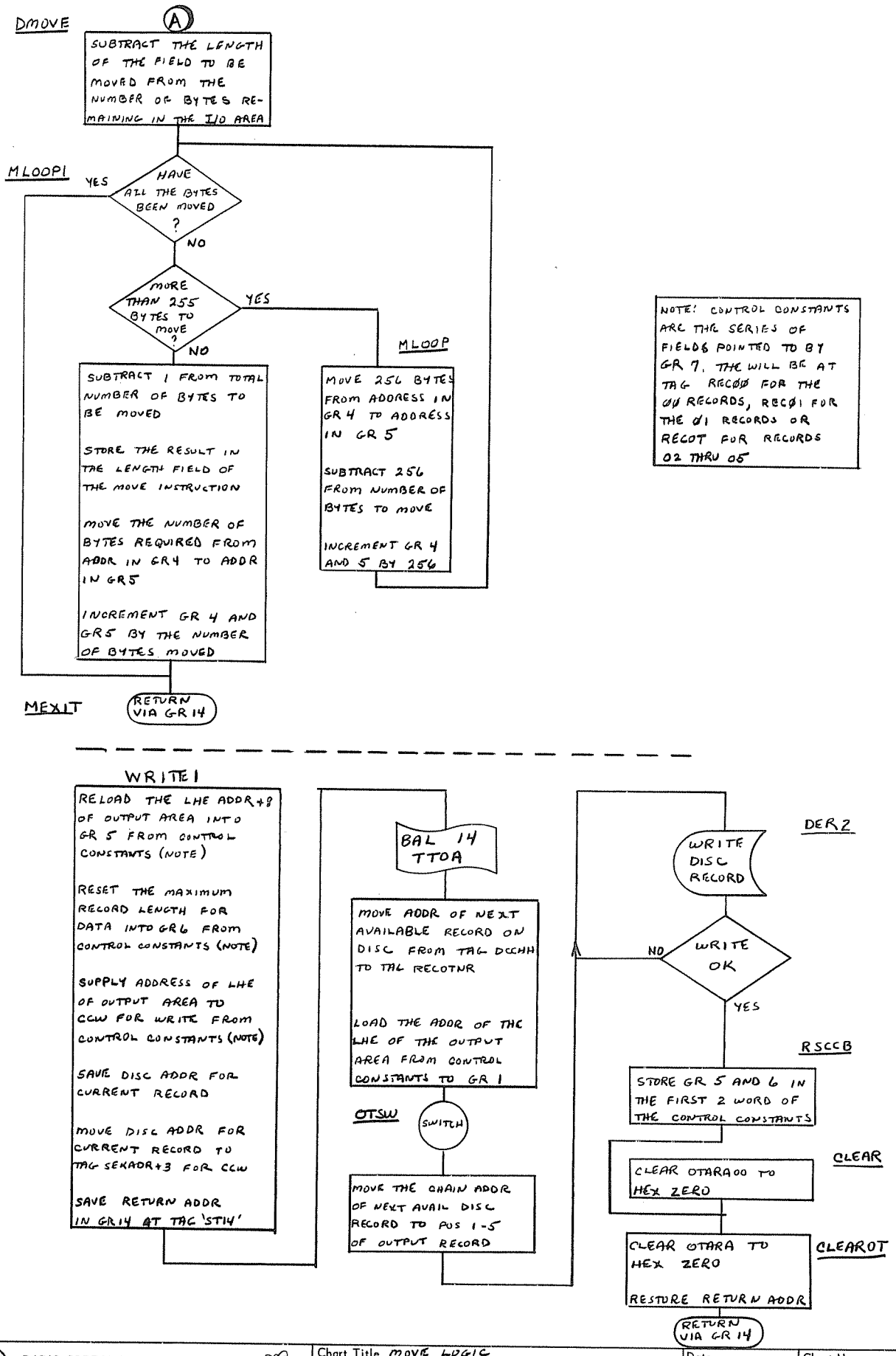
RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title: TEXT BLOCK (04) PROCESSING MODIFIER BLOCK (05) PROCESSING	Date	Chart No. 13 of 24
System Title: DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	Program Title	Revision: Letter	Date



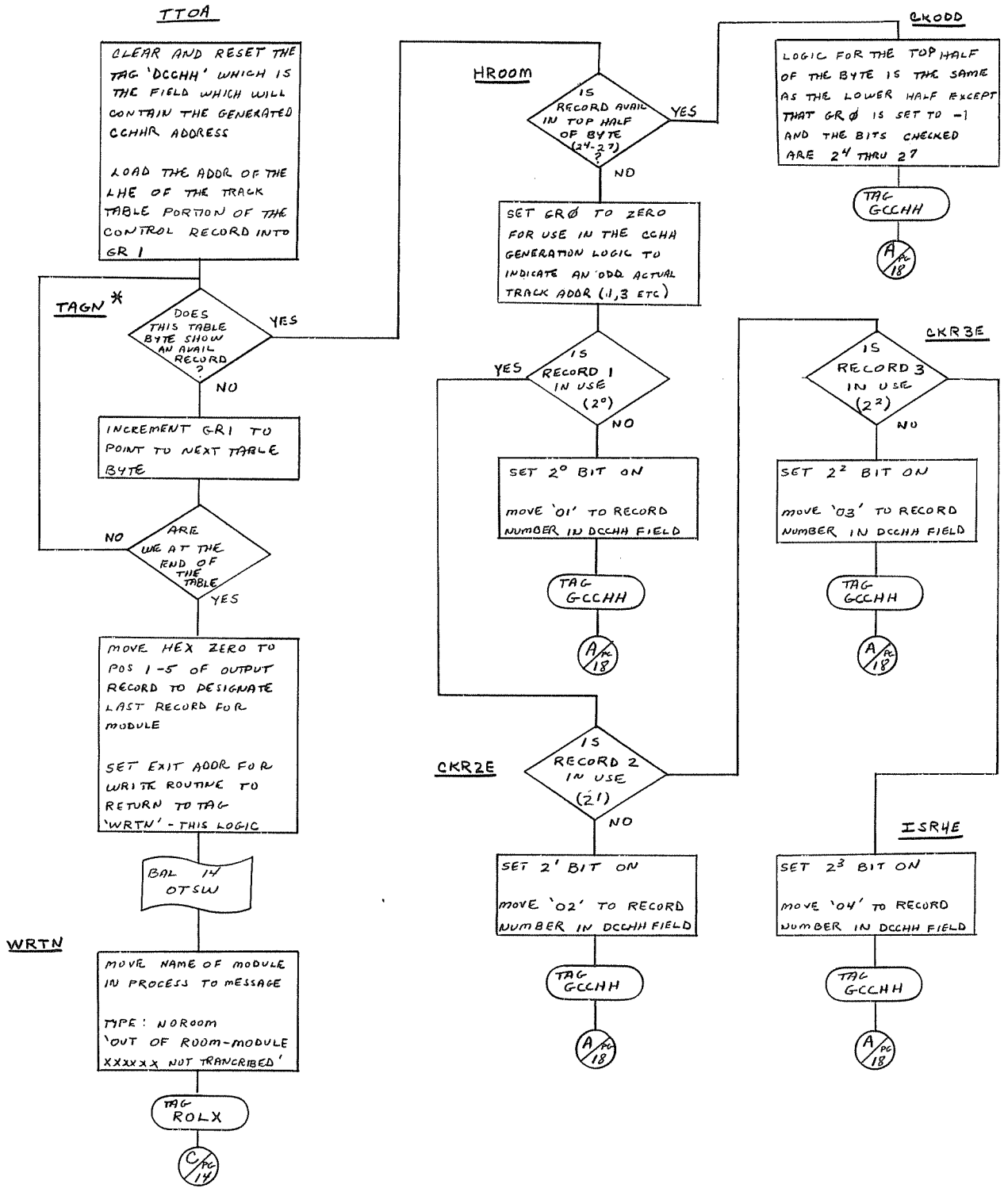
RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title: END OF JOB LOGIC	Date	Chart No. 14 of 24
System Title: DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	Program Title	Revision: Letter	Date



RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title MOVE LOGIC WILL RECORD TO BE MOVED FIT IN I/O	Date	Chart No. 15 of 24
System Title DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	Program Title SUB ROUTINE 'MOVE'	Revision Letter	Revision Date

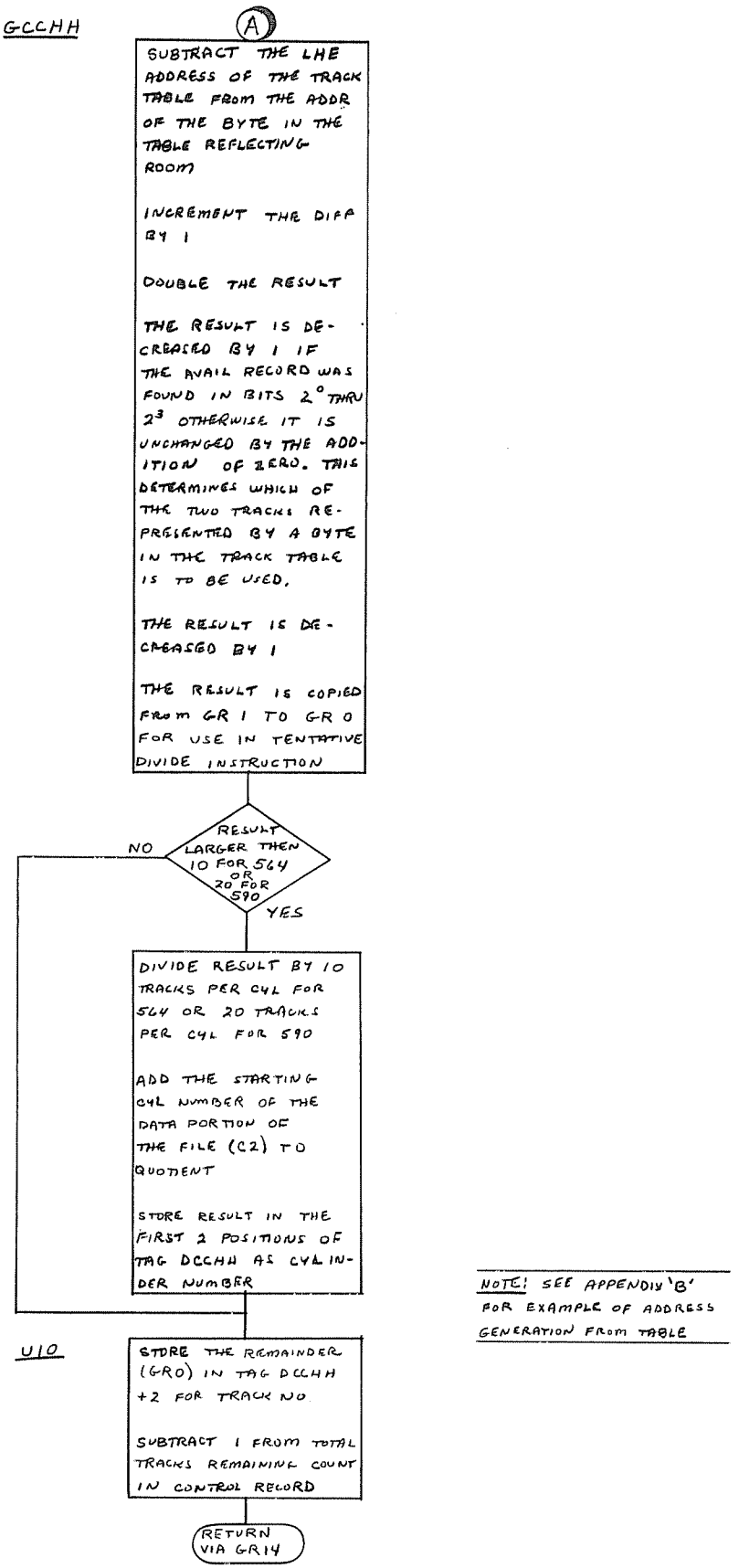


RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title MOVE LOGIC MOVE LOGIC FROM INPUT TO OUTPUT AND WRITE LOGIC	Date	Chart No. 16 of 24
System Title DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	Program Title	Revision Letter	Revision Date

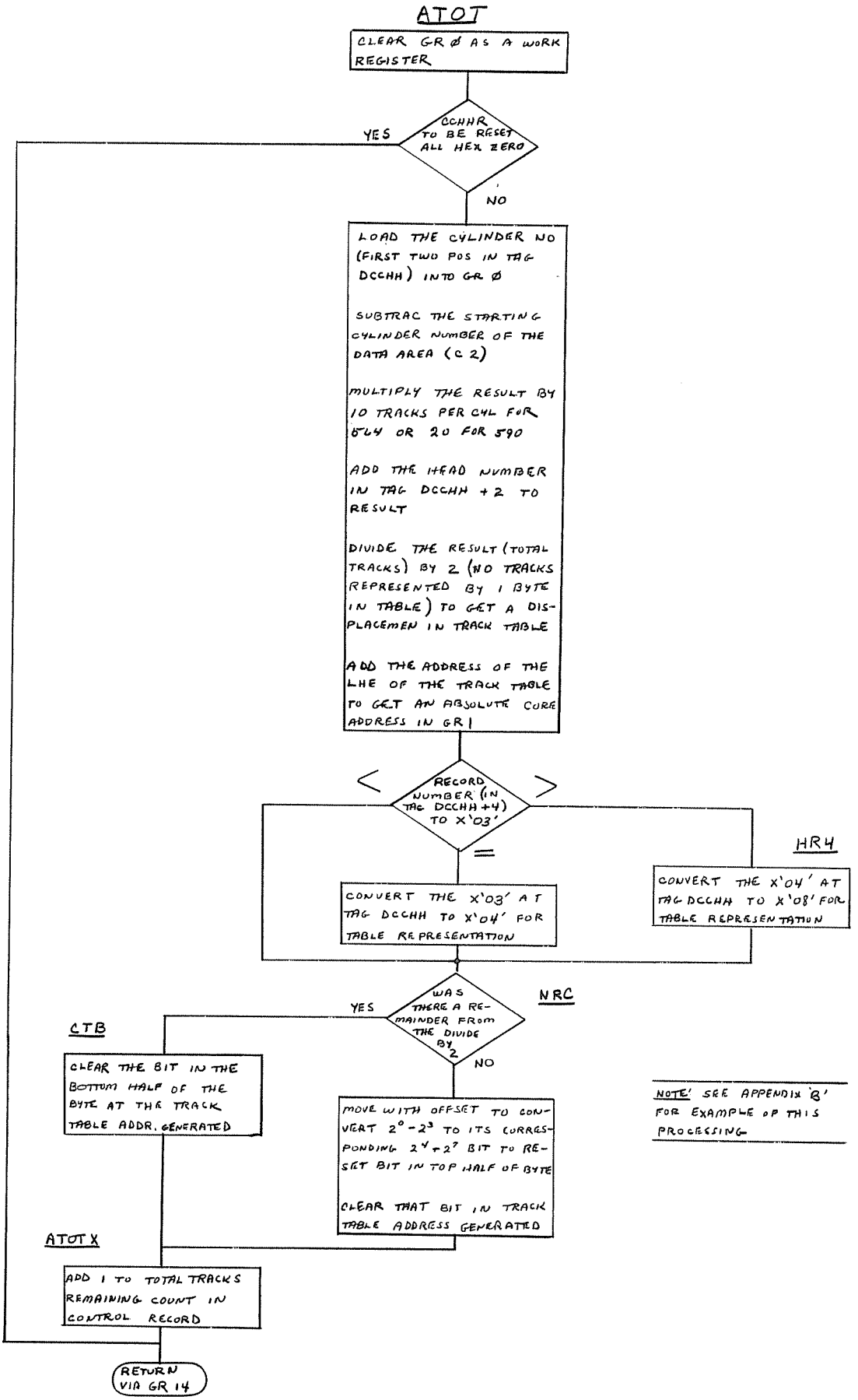


* FOR THE 70/590 THE BYTE IN THE TABLE IS CHECKED FOR HEX 'FF' TO DETERMINE IF ALL RECORDS IN THE TWO TRACKS REPRESENTED ARE USED. FOR THE 70/564 THE BYTE IS CHECKED FOR HEX '77'

RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title: TABLE TO ADDRESS LOGIC (TTOA)	Date	Chart No. 17 of 24
System Title: DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	Program Title: DETERMINE RECORD NUMBER	Revision: Letter	Date



RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title: TABLE TO ADDRESS LOGIC (TTOA)	Date	Chart No. 18 of 24
System Title: DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	Program Title: CALCULATE CYLINDER AND HEAD NO,	Revision: Letter	Date



RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title: RESET USED BIT IN TABLE BASED ON CCHHR ADDRESS (ATOT)	Date	Chart No. 19 of 24
System Title: DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	Program Title	Revision: Letter	Date

READØ

MOVE THE CCW TO READ INTO AREA 'OTARAØØ' TO THE CCW CHAIN

SAVE THE RETURN ADDR IN GR14 AT TAG 'IØ14ST'

DER3

READ THE 00 (DIRECTORY) RECORD — SEE NOTE 1

SEE NOTE 2 — READ OK — NO

YES

RDØX

RESET THE CCW IN THE CHAIN TO 'WRITE' FROM TAG 'OTARA'

MOVE THE CCHHR OF THE RECORD JUST READ FROM TAG 'SEKADR'+3 TO THE CONTROL CONSTANT GROUP DESIGNATE BY GR 7

RESTORE RETURN ADDR.

RETURN VIA GR14

NOTE 1

THE CCHHR OF THE RECORD TO BE READ MUST RESIDE AT TAG SEKADR+3 PRIOR TO 'BAL' INTO 'READØ' OR 'READ1'

NOTE 2

A BAL 14 TO TAG 'RTRY' IN THE GETEM MACRO MAINTAINS A COUNT OF THE NUMBER OF RETRIES. AFTER TWO ATTEMPTS, A MESSAGE IS TYPED TO THE OPERATOR

WRITEØ

NORMALLY THE CCW CHAIN CONTAINS A 'WRITE' CCW SO THE ADDR TO WRITE FROM IN THE CCW IS CHANGE TO THE ADDR SPECIFIED IN THE CONTROL CONSTANT AREA

SAVE THE RETURN ADDR IN GR14 AT TAG 'IØ14ST'

MOVE THE CCHHR ADDR OF THE LAST RECORD READ FROM THE CONTROL AREA TO TAG 'SEKADR' +3 FOR USE BY CCWS

DER4

WRITE THE 00 (DIRECTORY) RECORD

SEE NOTE 2 — WRITE OK — NO

YES

RWØX

RESET THE ADDR IN THE CCW TO WRITE FROM TAG 'OTARA'

RESTORE RETURN ADDR.

RETURN VIA GR 14

READ1

MOVE THE CCW TO READ INTO AREA 'OTARA' TO CHAIN

SAVE THE RETURN ADDRESS IN GR14 AT TAG 'IØ14ST'

DER5

READ THE 01 (INDEX) RECORD — SEE NOTE 1

SEE NOTE 2 — READ OK — NO

YES

RD1X

RESET THE CCW IN THE CHAIN TO 'WRITE' FROM TAG 'OTARA'

MOVE THE CCHHR OF THE RECORD JUST READ FROM TAG 'SEKADR'+3 TO THE CONTROL CONSTANT AREA DESIGNATED BY GR 7

RESTORE RETURN ADDR

RETURN VIA GR14

WRITEØ1

MOVE CCHHR ADDR OF THE LAST RECORD READ FROM THE CONTROL AREA TO TAG 'SEKADR'+3 FOR USE BY CCW

SAVE THE RETURN ADDRESS IN GR14 AT TAG 'IØ14ST'

DER6

WRITE THE 01 (INDEX) RECORD

SEE NOTE 2 — WRITE OK — NO

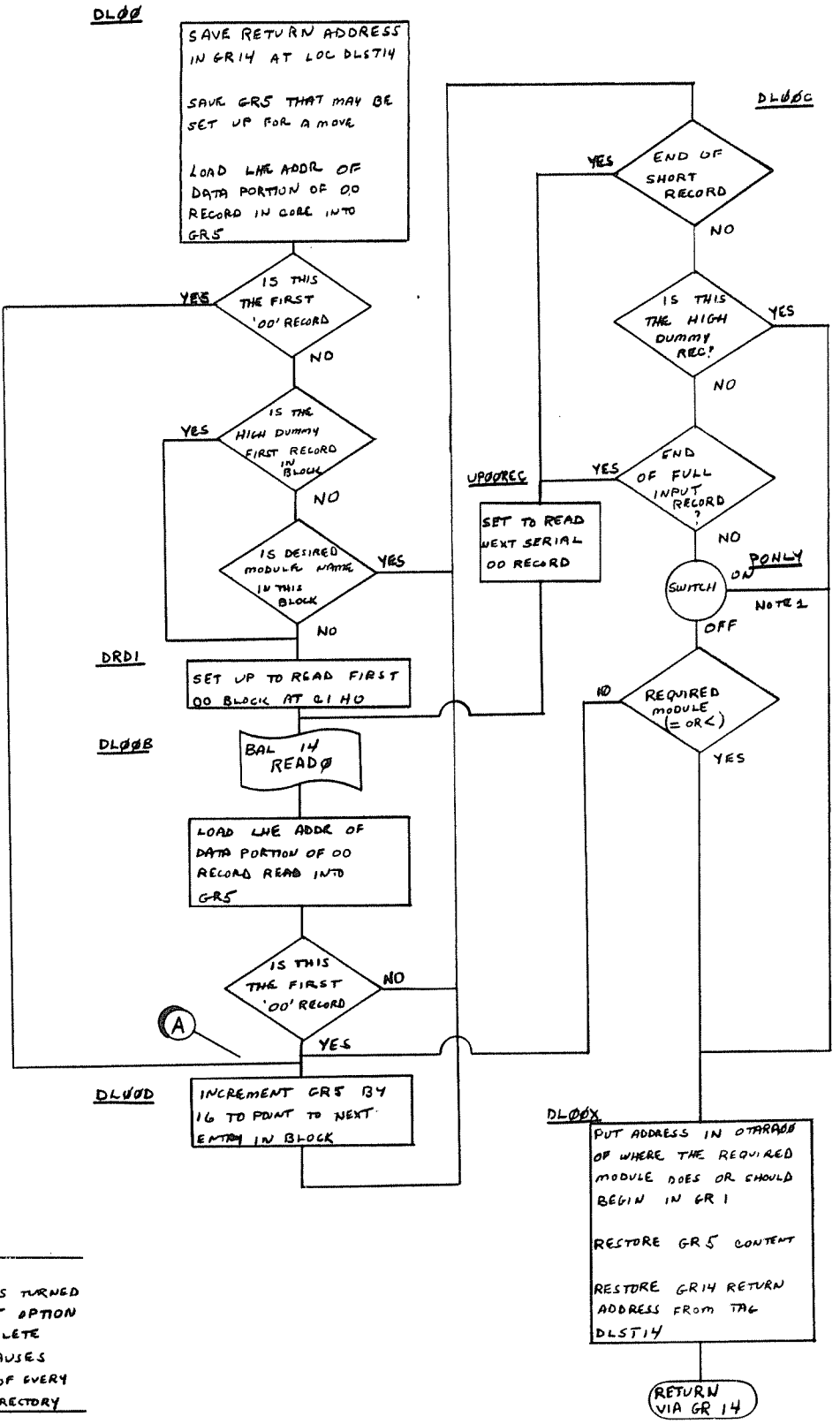
YES

RW1X

RESTORE RETURN ADDRESS

RETURN VIA GR 14

RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title: READ LOGIC FOR RECORD TYPE (00) USED TO REWRITE UPDATED 00 RECORD	Date	Chart No. 20 of 24
System Title: DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	~~Program Title~~: READ LOGIC FOR RECORD TYPE 01 USED TO REWRITE UPDATED 01 RECORD	Revision: Letter	Date



NOTE 1
THIS SWITCH IS TURNED ON FOR $PRINT OPTION RESPONSE TO DELETE MESSAGE. IT CAUSES SERIAL ACCESS OF EVERY ENTRY IN THE DIRECTORY

RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title: LOCATE MODULE NAME IN 00 RECORD	Date	Chart No.: 21 of 24
System Title: DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	Program Title: SUB MODULE	Revision: Letter	Date

CHASEØ1

STORE THE RETURN ADDR IN GR14 AT TAG 'CHST14'

SAVE THE CCHHR ADDR OF THE CURRENT 00 RECORD IN TAG 'SEKADR' AT TAG 'SEKADRS' TO BE RESET IN CHASE02 LOGIC

GR1 POINTS TO A DIRECTORY ENTRY IN OTARA00. MOVE CCHHR OF THE 01 RECORD FROM THAT ENTRY (FIELD 'D' APPDX 'A') TO SEKADR+3 FOR USE BY CCW

BAL 14 READ1

SET GR1 TO THE ADDR OF FIELD 'E' IN THE 01 RECORD JUST ACCESSED INTO AREA 'OTARA'

RESTORE RETURN ADDRESS

RETURN VIA GR14

CHASEØ2

STORE THE RETURN ADDR IN GR14 AT TAG 'RETØ2'

MODIFY CCW CHAIN TO 'READ' INTO AREA 'OTARA'

RDDET

READ INTO 'OTARA'

READ OK ?

NO

YES

RDOKDT

MOVE THE CCHHR OF THE RECORD LAST ACCESSED (INITIALLY THE ADDR OF THE 01 RECORD READ IN CHASEØ1 LOGIC) FROM TAG 'SEKADR'+3 TO TAG 'DCCHH' IN PREPARATION FOR ATDT LOGIC

BAL 14 ATDT

CHAIN ADDR OF LAST RECORD READ ALL ZERO ?

YES

NO

CHASEX

RESTORE THE CCHHR OF THE CURRENT 00 RECORD STORED IN CHASEØ1 LOGIC

RESTORE THE CCW IN CHAIN TO 'WRITE' FROM OTARA

RESTORE RETURN ADDR

RETURN VIA GR14

MOVE THE CHAIN ADDR FROM POS 1-5 OF CURRENT RECORD TO SEKADR+3 FOR USE BY CCW

RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title: LOCATE 01 RECORD BASED ON 00 POINTER CHASE DATA RECORDS FROM 01 TRANSACTION	Date	Chart No. 22 of 24
System Title: DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	Program Title	Revision: Letter	Date

SHIFTØØ

STORE THE RETURN ADDR IN GR14 AT TAG 'DLSTIN'

GR5 CONTAINS THE LHE ADDR OF THE DIRECTORY RECORD TO BE DELETED. COPY THIS ADDR INTO GR4 (GR5 IS THE 'TO' ADDR FOR 'move' LOGIC)

INCREMENT GR4 BY 16 TO POINT TO LHE OF NEXT DIRECTORY ENTRY (GR4 IS THE 'FROM' ADDR FOR 'move' LOGIC)

LOAD THE LHE ADDR OF THE DATA PORTION OF THE 00 RECORD (OTARA00+8) INTO GR1

ADD THE TOTAL NUMBER OF USED DATA BYTES IN THE RECORD

SUBTRACT THE ADDR OF THE LHE OF THE FIELD TO BE MOVED FROM THE RHE OF THE FIELD TO BE MOVED (GR1) TO GET THE NUMBER OF BYTES TO MOVE

SET GR6 TO 2000 TO INHIBIT THE MOVE LOGIC FROM WRITING A RECORD

STORE THE CALCULATED MOVE LENGTH IN THE SPEC PACKET FOR THE 'MOVE' LOGIC

BAL 14 MOVE

ADJUST THE RECORD LENGTH DOWN BY 16 FOR REMOVED ENTRY

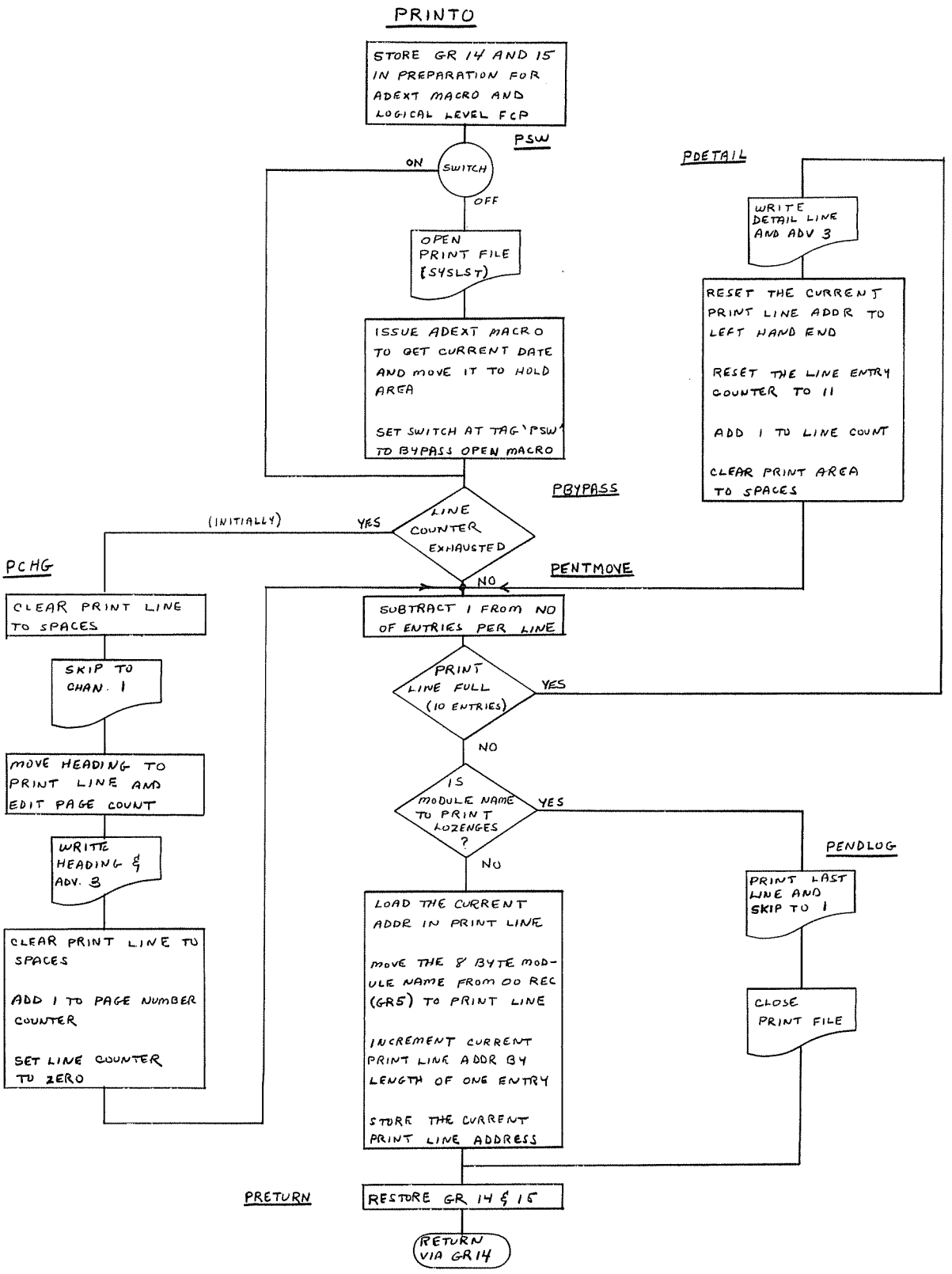
AFTER THE MOVE, GR 5 WILL BE POINTING TO THE LHE OF THE LAST ENTRY IN THE DIRECTORY BLOCK. BECAUSE THE BLOCK WAS SHIFTED LEFT 16 BYTES, THIS ENTRY WILL HAVE BEEN DUPLICATED. CHANGE THE LAST ENTRY TO HEX ZERO TO ERASE IT.

BAL 14 WRITEØ

RESTORE RETURN ADDR

RETURN VIA GR14

RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title: SHIFT 00 RECORD TO DELETE ENTRY	Date	Chart No. 23 of 24
System Title: DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	Program Title	Revision: Letter	Date



RADIO CORPORATION OF AMERICA ELECTRONIC DATA PROCESSING	Chart Title: MODULE NAME PRINT ROUTINE	Date	Chart No. 24 of 24
System Title: DISC OBJECT MODULE LIBRARY MAINTENANCE ROUTINE (DOMLMR)	Program Title	Revision: Letter	Date

Input Record To Output Record Conversion

See TOS Utilities Manual 70-35-302 (Object Module Library) for a description of the input tape.

To illustrate the conversion from tape Object Module Library (OML) format to disc OML format, charts have been intersperced with the narrative. The space between each dot represents two bytes. The tape and disc positions indicated are zero relative. A field with no tape position indicated in the 'From Tape' line indicates information generated by the program. A layout with no 'To Disc' line indicates that the particular record type can be located anywhere within the data bytes alloted in each disc record. The first 8 positions reflect the five position CCHHR address of the next logical record for the module, followed by a three position hexidecimal count field of the number of data bytes that are used.

The basic processing is as follows:

Record type $(00)_{16}$ (Descriptor Block) - fixed length of 16 bytes on disc.
Each 16 byte entry is merged into a sequential location in one of the records located in cylinder 1 track 0 thru 8.
Record type $(01)_{16}$ (Index Block) - starts a new record in the data portion of the file (note 1).
Record type $(02)_{16}$ (Descriptor Block) - starts a new record in the data portion of the file (note 1) and is immediately followed on the output record by the next input transaction $(03)_{16}$, $(04)_{16}$ or $(05)_{16}$.

Note 1: The data portion of the file is cylinder two track zero to end of extent for 70/590 and 70/564. This is the area covered by the track table contained in the DOMLMR control record. Cylinder one tracks zero thru eight contains the Object Module Directory, cylinder one track nine is the DOMLMR control record. Cylinder one tracks eleven thru twenty are not used on the 70/590.

Record Size: 70/590 is 1688 byte records - 4 per track - 1680 data portion.
70/564 is 1129 byte records - 3 per track - 1121 data portion.

Object Module Directory Block $(00)_{16}$

Disc Record Layout

	A	B	C	D	E
From TAPE			9 16		
To DISC	0 4	5 7	8 15	16 20	21 23
Size	5	3	8	5	3

Field Contents

A. CCHHR of next Object Module Directory Block (next sequential record in cylinder one).

B. Three position byte count for this block - represents the sum of the 16 byte entries only. This field is calculated in the Object Module Directory Block Logic $(00)_{16}$.

Fixed length entries of 16 bytes - one per module.

C. Eight position module name.

D. CCHHR of the corresponding Index Block in the data portion of the field. (established when the Index Block - $(01)_{16}$ is read from tape.)

E. Three position displacement of the Index Block within the record specified in field 'D'. (zero in all cases because each Index Block starts a new record)

General:

1. The total record length is 1129 bytes (16 byte entries X 70 entries per record) + 8 control characters + 1 unused byte at the end of the record) for 70/564. For the 70/590 the total record length is 1688 bytes (16 byte entries X 105 entries per record) + 8 control characters.

2. DOMLMR uses tracks 0 thru 8 of cylinder 1 to contain the Object Module Directory.

3. Entries do not overflow records.

4. The first entry of the first record contains - 'OMLU 22 10 YYJJJ' where YYJJJ is the year and julian date of the last reference to the file by DOMLMR.

5. The last entry of the Directory contains a module name of lozenges, a CCHHR of 'CCHHR' and a displacement of hex zero except for 2^7 of the high order position.

Index Block (01)16
Disc Record Layout

	A	B	C	D	E	F	G	H	I	J	K
From TAPE				11 18				22 57	45 74	55 58	34 96
To DISC	04	57	89	11 07	12 82	23	26	22 79	33 07	34 81	44 92
Size	5	3	2	8	5	3	1	3	8	4	8

	L	M	N	
From TAPE	33 18	55 78	57 09	72 01 98
To DISC	55 07	55 89	67 10	72
Size	8	2	12	

Index Block $(01)_{16}$
Disc Record Layout

Field Contents

A. CCHHR of first detail record for this module. Same as field 'E'. Established when the $(02)_{16}$ record is read from tape.

B. Byte count for this block - represents size of data portion only.

C. Byte count for the record.

D. Module name.

E. CCHHR of first detail record for this module. (Same as field 'A') Established when $(02)_{16}$ record is read from tape.

F. Displacement of data in the record located at the CCHHR address indicated in field 'E'. (zero in all cases because each Object Module descriptor Block $(02)_{16}$ starts a new record).

G. Unused - pad with zero.

H. Module length.

I. Extern name.

J. Starting address.

K. DDNAME (for include)

L. OMNAME (for include)

M. Unused - set to zero.

N. First 12 byte entry name (8 bytes) and starting address (4 bytes). This field is repeated for each entry, extern and or common for this module.

General:

1. Each Index Block begins a new record.

2. The Index Block is variable length as determined by number of entries, externs and common. (total entries, externs, and common X 12 Bytes per entry + 50).

3. The address of the Index Block record is determined by the table search logic in the TTOA logic.

Object Module Descriptor Block $(02)_{16}$
Disc Record Layout

	A	B	C	D	E	F	G
From TAPE				0	1	2 - 5	6 - 13
To DISC	0 - 4	5 - 7	8 9	10	11	12 - 15	16 - 23
Size	5	3	2	1	1	4	8

Field Contents

A. CCHHR of next record for this module or hex zero if this is the last record for the module.

B. Block length - initially set to the maximum data bytes to reflect a full record in the $(02)_{16}$ processing. Reset to adjusted length when the next $(02)_{16}$ is read or the end of the input tape is read in the End of Job Logic.

C. Record length - '000E' for $(02)_{16}$ record.

D. Block Code $(02)_{16}$

E. Type of block that follows (See Utilities Manual for code explanation.)

F. Hex zero - reserved for future use.

G. Module name.

General:

1. Each Object module Description Block begins on a new record. It will be immediately followed by one of the other record types (03_{16}-05_{16})

2. The CCHHR address of this record is contained in the corresponding Index Block $(01)_{16}$ fields A and E.

EXTRN Block $(03)_{16}$

Disc Record Layout

	A	B	C	D	E	F	G
From TAPE		0	1	8 11	1 12 — 1 9	2 0	2 2
To DISC							
Size	2	1	1	4	8	1	2

Field Contents

A. Length of $(03)_{16}$ record - starting in tape relative position 12 is a variable number of 11 byte EXTRNS. This length is computed in the 03 logic by multiplying the number of these 11 byte entries by 11 and adding 6 additional bytes for the fixed length portion of the disc record.

B. Block Code - $(03)_{16}$

C. Block Subcode (See Utility Manual for code explanation)

D. Filler - not used.

E. EXTRN

F. Type Code

G. ESID

General

1. Fields E, F and G represent one EXTRN. These fields are repeated for each EXTRN.

2. The EXTRN block may be located anywhere in the data portion of a disc record and may overflow records.

Text Block $(04)_{16}$

Disc Record Layout

	A	B	C	D	E
From TAPE		0	1	4 7	2 0.....N
To DISC					
Size	2	1	1	4	up to 1044

Field Contents

A. Length of $(04)_{16}$ record - this size is calculated by adding 6 bytes to the 'Block Byte Count' field positions 2 and 3 of the $(04)_{16}$ record.

B. Block code $(04)_{16}$

C. Block Subcode (See Utilities Manual for code explanation.)

D. Load address of next block.

E. Text - variable size - up to 1044 bytes.

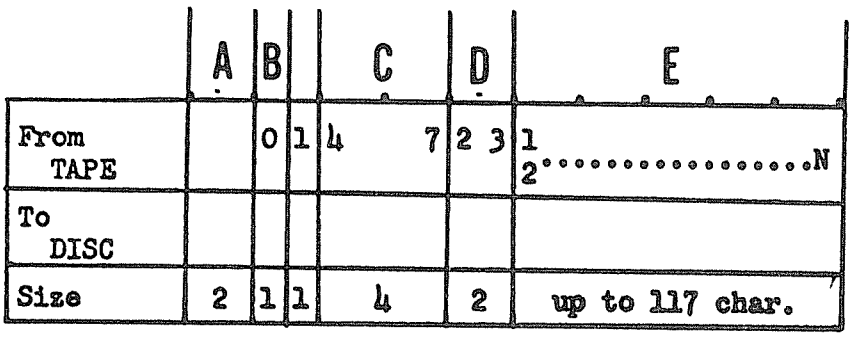
General

1. The move logic will insure that fields A, B and C will fit on the end of the current output record when field 'A' is moved. To accomplish this the current output record must have 10 or more bytes remaining prior to the 'A' field move. If it does not have sufficient room, the length of the current output record (OTARA + 6 and 7) is modified, the record is written to disc and the 'A' field is placed in the next record starting in OTARA + 8.

2. The 04 record may reside anywhere in the data portion of a disc record and may overflow records.

Modifier Block $(05)_{16}$

Disc Record Layout



	A	B		C	D	E
From TAPE		0	1	4 7	2 3	1 2 ..... N
To DISC						
Size	2	1	1	4	2	up to 117 char.

Field Contents

A. Length of $(05)_{16}$ record - this size is calculated by subtracting 4 from the length of the tape record read.

B. Block Code $(05)_{16}$

C. Block Subcode (See Utilities Manual for code explanation.)

D. Load address of next block.

E. Modifier count.

F. Modifiers - 10 bytes each. The length of this variable portion of the record is calculated by subtracting 12 from the length of the input record. (See Utilities Manual for contents of modifier record.)

General

1. The 05 record may reside anywhere in the data portion of disc record and may overflow records.

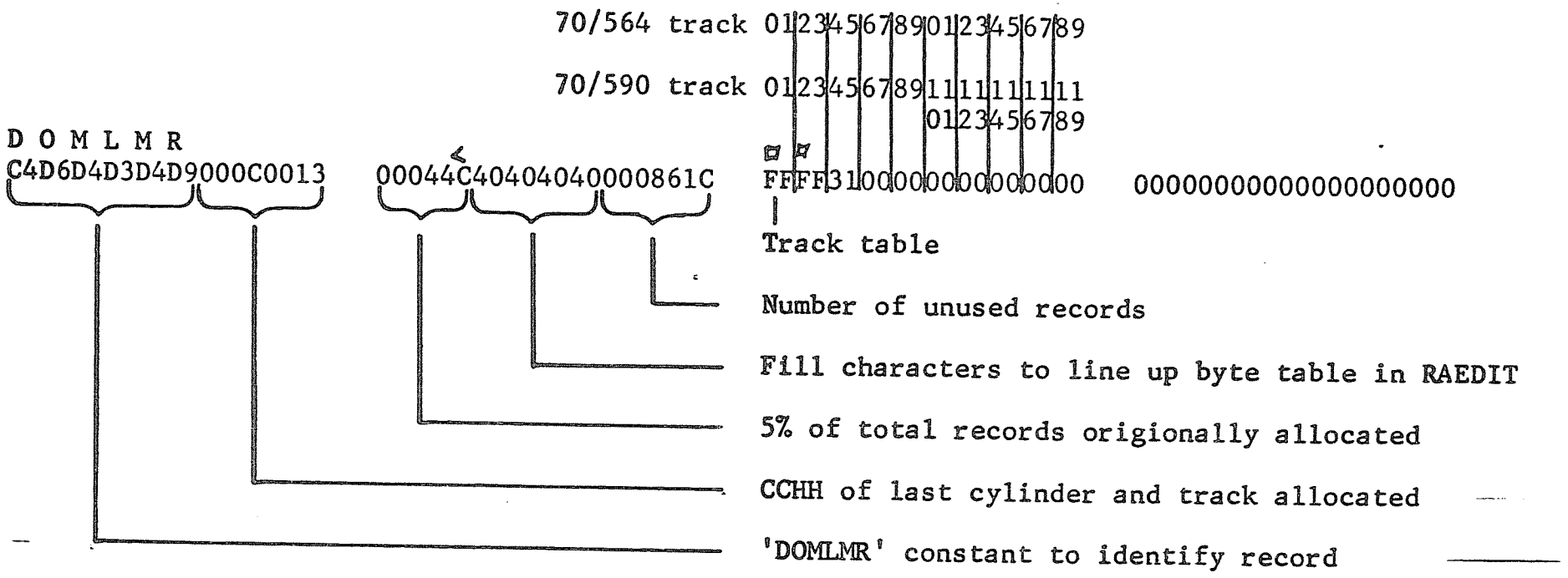
2. Fields A thru E may not be split, the balance of the input record may be.

CONTROL RECORD FORMAT

THIS RECORD IS CONTAINED ON CYLINDER 1, TRACK 9, RECORD 1

FIELD	POSITION	SIZE	DESCRIPTION
1	0-5	6	RECORD IDENTIFIER (DOMLMR)
2	6-9	4	CCHH OF LAST TRACK IN EXTENT
3	10-12	3	5% OF ORIGINAL NUMBER OF RECORDS AVAILABLE. PROGRAM WILL TYPE WARNING MESSAGE WHEN FIELD 5 BECOMES EQUAL TO OR LESS THAN THIS AMOUNT. (PACKED)
4	13-16	4	NOT USED - FOR RAEDIT ALIGNMENT ONLY.
5	17-19	3	NUMBER OF UNUSED RECORDS IN DATA PORTION OF FILE (PACKED).
6	20-2019	2000	ONE BYTE REPRESENTS 2 TRACKS. THE TOP HALF OF THE TYPE (2^4-2^6) REPRESENTS RECORDS 1 THRU 3 (RELATIVE) FOR THE ODD NUMBERED TRACKS. THE BOTTOM HALF OF THE BYTE (2^0-2^2) REPRESENTS RECORDS 1 THRU 3 (RELATIVE) FOR THE EVEN NUMBERED TRACKS. BITS 2^3 AND 2^7 REPRESENT RECORD 4 FOR THE 70/590. A 1 BIT INDICATES RECORD IN USE, A 0 BIT INDICATES RECORD NOT IN USE.

The following example illustrates the DOMLMR control record.



Track Table;
The 'FFFF' in the first two positions indicates that records 1 thru 4 of tracks 0 thru 3 of cylinder 2 are used for a 70/590. The same indication for a 70/564 would be '7777' because only records 1 thru 3 are used.

The '31' indicates:

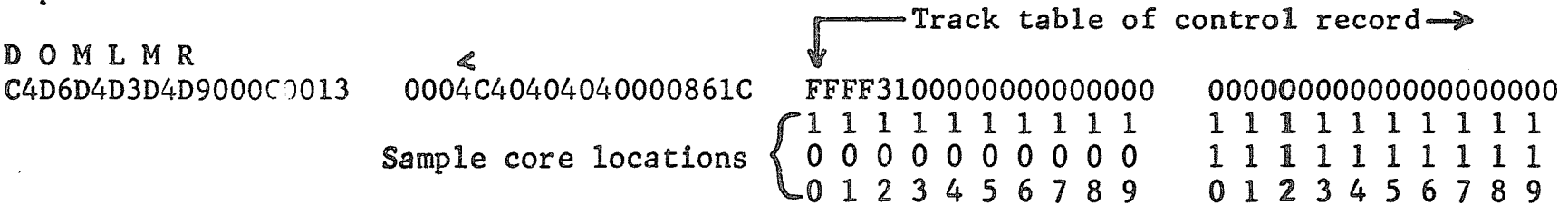
2^7 = 0 -record 4 not used (70/590 only)
2^6 = 0 -record 3 not used
2^5 = 1 -record 2 used — Track 4
2^4 = 1 -record 1 used

2^3 = 0 -record 4 not used (70/590 only)
2^2 = 0 -record 3 not used
2^1 = 0 -record 2 not used — Track 5
2^0 = 1 -record 1 used

Starting with the table location corresponding to the ending CCHH+1 thru table position 2000 all locations are set to 'FF' for 70/590 or to '77' for 70/564 by PREOML. This is to prevent generating an address beyond the allocated area.
Each decade of the track table represents 1 cylinder for the 70/590. Each half decade represents a cylinder for the 70/564. The table starts at cylinder 2.

The following example illustrates the use of the DOMLMR control record for disc address generation.

A partial RAEDIT of a DOMLMR control record:



Address generation is accomplished in the Table to Address (TTOA) logic in DOMLMR as follows:

```
 102  address of the first table position
      that does not contain 'FF' or '77'.
-100  subtract starting address of table.
   2
+  1  add 1.
   3
   3  double the result.
   6
+ (1) this value is -1 if the unused bit
      is located in 2^4 thru 2^7 else zero.
   5
-  1  subtract 1.
   4  = track number - if over 10 or 20,
      divide by 10 or 20 for cylinder num-
      ber, in which case, the remainder will
      be the track number. The result is
      added to the base address of C2, H0
      to arrive at the actual address.
```

Reset unused bit in table based on CCHHR address. This is accomplished in the Address To Table (ATOT) logic in DOMLMR.

```
Example: 0002000403
                C C H H R
         -   2  subtract starting cylinder no.
          0000
         X   20  multiply by tracks per cylinder
                 (10 for 70/564 or 20 for 70/590)
          0000
         + 0004  add the head number.
        2)0004   divide by 2 tracks per byte in
          0002   table
         + 100   add starting address of table
           102   -effective table address.
```

NOTE Next available Record is determined by a test under mask (TM) starting with 2^4- 2^7 then 2^0- 2^3. Once an open bit is detected, it is turned on by an OI instruction to indicate used. See TTOA logic.
An XC instruction is used in ATOT to reset used bits to zero.